GENERAL DIGITAL INDUSTRIES

SOFTWARE TECHNOLOGY TESTBED "SOFTPANEL" PROTOTYPE

Contract # NAS8-37680

FINAL REPORT

May 1988 - February 1991

Submitted To: Bob Stevens/EB42
CC: Chris Hauff/KA40

Date: February 19, 1991

SUMMARY

This document is the Final Report related to NASA/MSFC contract NAS8-37680. The following is a quantitative summary of the major accomplishments performed between May 1988 and Feb. 18, 1991 on the "SoftPanel Prototype" contract.

- **1.0 ANALYSIS PERFORMED**
  - 1.1 Approximately 16 Unique Areas of Study
- **2.0 CONTRACT SUPPORT**
  - 2.1 Meetings/Telecons/Videocons Attended
    - 2.1.1 Approximately 450 On-Going Meetings
    - 2.1.2 Approximately 137 Special Meetings
  - 2.2 Reviews Supported
    - 2.2.1 Approximately 24 Reviews Supported
  - 2.3 Trips Supporting Contract
    - 2.3.1 Approximately 27 Trips Taken
  - 2.4 Documents Reviewed
    - 2.4.1 Approximately 869 Documents Formally Reviewed
    - 2.4.2 Approximately 62 Documents FYI
    - 2.4.3 Approximately 1,037 Documents Stored in Library
  - 2.5 RIDs Produced
    - 2.5.1 Approximately 2100 RIDs Produced
  - 2.6 Classes/Seminars Attended
    - 2.6.1 Approximately 11 Classes Attended
    - 2.6.2 Approximately 10 Seminars Attended
- **3.0 HARDWARE PURCHASED**
  - 3.1 MicroVax Hardware Upgrade
  - 3.2 Apollo Disk Upgrade
- **4.0 HARDWARE/SOFTWARE DEVELOPED**
  - 4.1 Hardware Developed
    - 4.1.1 Aquarium System
  - 4.2 Software Developed
    - 4.2.1 AG Control Diagrams and Ada Code for Aquarium System
    - 4.2.2 SoftPanel Prototype
      - 4.2.2.1 User I/F using DataViews
      - 4.2.2.2 User I/F for ECLSS Testbed using TAE+/ XWindows
- **5.0 LEVEL II SUPPORT**
  - 5.1 Level II SIM Support
    - 5.1.1 Lead on SIM Task Team SIM Architecture group
  - 5.2 Standards
    - 5.2.1 SMADR Officially Signed-off on By NASA/MSFC
    - 5.2.2 MASM Presently Under Review by NASA/MSFC
    - 5.2.3 Spiral Model Development
  - 5.3 Risk Management Assessment
    - 5.3.1 Risk Management Plan Developed
    - 5.3.2 Risk Management Assessment of Space Station Software
    - 5.3.3 Assessment of Needed/Available Technology Base

REPORT

The report that follows is an overview of the main areas of analysis and development over the duration of the "Software Technology Testbed "SoftPanel" Prototype" contract (May 1988 - February 1991) under contract number NAS8-37680. Areas of support and emphasis were tailored over the life of the contract to cover the specific needs of that period.

1.0 ANALYSIS PERFORMED

GDI concentrated on a variety of areas of analysis and design over the life of the contract.

1.1 Approximately 16 Unique Areas of Study

The main areas of analysis performed over the life of the contract are listed below:

a. Analysis of the advantages and disadvantages of using Ada for the development of real-time control systems for the Space Station. This included analysis of the real-time development environments available and analysis of the Lynx operating system in fulfilling the POSIX standards.

b. Analysis of the functionality of the Application Generator (AG) through the development of the Aquarium Control System.

c. Analysis of the User Support Environment (USE) criteria (DR SY-45) through the development of the SoftPanel Prototype on the SUN workstation using DataViews/TAE+/XWindows.

d. Analysis of SSE tools and procedures which are to be used for the development of ground/flight software for the Space Station. This included analysis of the Rational computer as a suitable development environment, analysis of the MAC, PS/2 and the Apollo as the suitable workstations, and analysis of the VAX mainframe and environment as a suitable host to drive the entire development and testing environment.

e. Analysis of the usefulness of the CBATS tutorial (an Ada tutorial package). Investigated its ease of use and functionality in comparison with how it allowed the user to efficiently learn Ada.

f. Analysis of Interleaf to see if it served as an adequate development environment for documentation. This was not only concerned with the development of the documentation, but also the adherence to specified standards, the ability to maintain adequate configuration management, and the ability to perform quality assurance of the documentation.

g. Analysis of the Integration, Test and Verification (IT&V) process of the Space Station. This included participation of such groups as the Integration, Test and Verification Environment (ITVE), Data Management System (DMS) Kits, Ground Support Environment (GSE) and Simulations groups. This analysis involved participation in many reviews and working group meetings to aid in the process of refining the integration of the testing and verification environments.

h. Analysis of the DMS on-orbit flight architecture. This included analysis and constant trade between the software requirements and code needs versus the available processor resources that could be provided due to power, weight and volume constraints. This also required analysis of non-DMS components to be used in the system, such as firmware controllers and the sensors/effectors.

i. Analysis of the simulation architecture as it pertained to the overall testing philosophy of the Space Station on the ground. This included a lead role in establishing a Level II baseline of the simulation architecture, and performing analysis of the existing Level II standards that were available for the purposes of revision to bring the various Work Packages into line with an integrated simulation and testing philosophy.

j. Analysis and updating of the Software Management and Requirements Document (SMADR) from the LAB level to the center-wide document that it has become. Also, developed the Methods and Standards Manual (MASM).

k. Analysis of the significant software differences between the Space Shuttle and Space Station software development efforts and design. A white paper was generated outlining the significant differences between these two software development efforts. This included an in-depth analysis of specific systems which were related between the two programs.

l. Analysis of the On-Orbit Verification & Validation (V&V) process. This included the discussion and addition of requirements to the Level II documentation to outline a complete plan for the migration of software from the ground to on-board.

m. Analysis of the DMS Command and Control architecture in conjunction with the timing and RODB/MODB requirements. This including making sure the tiered command structure of the DMS software was adequate to fulfill the needs of the various systems, real-time or not.

n. Analysis of the WP1 software sizing activity which analyzed the software development effort versus the amount of sensor/effectors identified in the system. This also looked into the necessity of those various modules of code in comparison to fulfilling the overall requirements of the identified systems.

o. Analysis of identified risks and development of a Risk Management plan for the Space Station. Applied the Risk Management plan to the Station to identify the risks. Also made an assessment of the available technology base as related to the risk needs of the Station.

p. Analysis of the RID process resulted in the development of an automated RID generating process which was designed and developed using SMALLTALK, an object oriented designed language. This P.C. based windowing function was designed to date, document and electronically generate RIDs and to alleviate problems encountered using the current manual process.

2.0 CONTRACT SUPPORT

The main emphasis of GDI's contract was to provide software support to MSFC/EB42 for the Space Station program in whatever areas were needed. This included analysis of various systems, modules and IT&V philosophies to insure the proper development and testing of Station related software. To perform this task, it was necessary to perform documentation reviews and attend technical meetings as outlined in the sections to follow. In conjunction with this support, many working papers and technical reports were generated in order to keep EB42 abreast of the activities that GDI participated in during the contracted period. This listing is included as Appendix A of this document.

2.1 Meetings/Telecons/Videocons Attended

To maintain current knowledge of Space Station events as they happened, it was necessary to attend many meetings including telecons and videocons. This aided in the analysis of the various systems as they matured through their life-cycles.

2.1.1 Approximately 450 On-Going Meetings

There were approximately 17 meetings that were considered on-going from the stand-point that they occurred on a weekly, bi-weekly or monthly basis. Over the life of the contract, GDI attended approximately 450 of these meetings. The on-going meetings were made up of the following:

1) Weekly SSE Pre-Board Meetings
2) Bi-Weekly SSE CCB Meeting
3) Bi-Weekly Project Status Meeting
4) Bi-Weekly DMS Kits Telecon
5) SSFP Simulation Task Team Telecons and Videocons
6) Aquarium System Monthly Progress Meeting
7) Monthly DMS Status Meeting
8) DMS Performance Telecons
9) MSFC/Boeing Technical Interchange Meetings (TIMs)
    a) Ground Support Environment (GSE)
    b) Integration, Test and Verification Environment (ITVE)
    c) Simulations
    d) Data Management System (DMS)
    e) Structures and Mechanisms
10) Systems Environment Working Group (SEWG) Telecons
11) Application Generator (AG) User's Working Group Telecon
12) MSFC/WP01 Software Review Board
13) Simulation Interface Buffer (SIB) Forum

2.1.2 Approximately 137 Special Meetings

GDI participated in approximately 137 meetings which were classified as special meetings. Special meetings are classified as those meetings which were important for GDI to attend in support of the contract, but were not considered a part of the meetings listed in section 2.1.1. Listed below are examples of some of the meetings attended which fall under this classification:

1) On-Orbit V&V meetings
2) DMS Command and Control meetings
3) RODB/MODB Design analysis meetings
4) Progress Review presentations (SRR, PDR, CDR)
5) Project Orientation meetings (UAH tasks)
6) Working Group Meetings (AG, USE, DMS, etc.)
7) Trip Related meetings (DMS, USE, SIMs, etc.)

2.2 Reviews Supported

To adequately keep up with the progress related to the modules and systems of the Space Station, it was necessary to actively participate in the reviews of products produced by WP01 and other SSFP centers. The review of the requirements and design was pertinent to the analysis of the WP01 design.

2.2.1 Approximately 24 Reviews Supported

In support of maintaining up-to-date information as it related to the WP01 design, GDI participated the major Station reviews. GDI participated in approximately 24 of these reviews which are listed as follows:

1) WP01 Delta PRR
2) WP01 Software PRR
3) DMS/OMS SRR/SDR
4) SSE PDR
5) DMS PDR #1 Long-Lead Hardware
6) SIB Detailed Requirements Review (DRR)
7) DMS PDR #2 Software
8) SSE PDR DIR #2 (DRLI 59)
9) DMS PDR #3 System
10) OMA PDR
11) WP01 Distributed System PDR/Software SRR
12) DMS SDR/SRR
13) WP02 Integrated PDR
14) SSFP UIL Spec. Review (DRLI 99)
15) ITVE SDR/SRR
16) ITVE PDR
17) WP01 LAB PDR
18) DMS Firmware SRR
19) SSE DIR #3
20) DNSIM/ALEPS PRR
21) SIB Integrated Detailed Design Review (IDDR) #1
22) ITVE IDDR #1
23) ISPDR
24) WP01 GSE SRR

2.3 Trips Supporting Contract

There were numerous trips taken in support of the contract. Trips were only authorized for very important meetings/working groups required to support the Space Station analysis effort.

2.3.1 Approximately 27 Trips Taken

There were approximately 27 trips taken including various reviews, working group meetings, workshops. The trips taken as part of the Space Station contract are listed as follows:

1) USE Working Group (WG)/Boulder, CO
2) SSE SDR/Lockheed/JSC
3) USE WG/JSC
4) DMS USE SRS/GSFC
5) USE WG/KSC
6) DMS/OMS SDR/SRR/JSC
7) DMS PDR #1/IBM/JSC
8) DMS WG/MDSSC/JSC
9) SSE DIR #1/Lockheed/JSC

10) SIB DRR/Lockheed/JSC
11) DMS WG/MDSSC/JSC
12) SSE DIR #2/Lockheed/JSC
13) SSE DIR #3/Lockheed/JSC
14) UIL Spec. Review/JSC
15) SIM Task Team Meeting/Reston
16) SIM Task Team Meeting/Reston
17) SIM Task Team Meeting/JSC
18) SIM Task Team Meeting/Reston
19) DMS PDR #1 (Summary Presentation)/IBM/JSC
20) DMS PDR #2/MDSSC/JSC
21) SMAP Workshop/San Diego, CA
22) DMS PDR #3 (Summary Presentation)/MDSSC/JSC
23) DMS PDR #3/MDSSC/Huntington Beach, CA
24) ITVE SDR/SRR/MDSSC/JSC
25) ITVE PDR (Summary Presentation)/MDSSC/JSC
26) ITVE PDR/MDSSC/JSC
27) ITVE/SIB IDDR #1/MDSSC/Lockheed/JSC

2.4 Documents Reviewed

In support of the Space Station project, GDI was required to review and maintain many documents throughout the life of the contract. The review of these documents was considered part of the analysis of the various systems and elements of the Space Station.

2.4.1 Approximately 869 Documents Formally Reviewed

In support of requirements and design analysis, there were approximately 869 documents reviewed. The majority of the documents that were reviewed were obtained as part of the requirements and design reviews that were held over the course of the contract. In addition to these documents were such reviewable items as white papers, CRs, and feasibility studies.

2.4.2 Approximately 62 Documents FYI

In addition to the reviewed documents, there were approximately 62 documents that were received and read as For Your Information (FYI). These documents mainly consisted of status reports outlining the progress of certain systems in the program.

2.4.3 Approximately 1,037 Documents Stored in Library

Throughout the life of the contract, GDI maintained a software library of all of the documents that were received that related to the Space Station. As a result of this effort, GDI has approximately 1,037 documents in its Space Station related software library. A listing of the documents that are contained in the library was delivered to EB42 on 1/31/91 under the title "SSFP Documentation Library" (WP-2210/0022/002/00). This serves as the complete listing of the documents in the library as of 2/19/91.

2.5 RIDs Produced

In support of the many documentation reviews that GDI attended as noted in section 2.2, there were many RIDs produced from formal review of the documentation.

2.5.1 Approximately 2100 RIDs Produced

In support of the 24 major reviews and other smaller review cycles that GDI participated in, there were approximately 2100 formal RIDs written against system documentation. These documents varied over many systems as well as many work packages.

2.6 Classes/Seminars Attended

GDI attended several classes/seminars in support of the contract. These were attended in order to expand GDI's technical expertise in areas pertaining to the contract. The seminars and classes were normally reported on if information pertinent to the contract was obtained.

2.6.1 Approximately 11 Classes Attended

There were several classes which were offered which were relevant to the expansion of knowledge as it related to the contract. GDI attended approximately 11 classes of this type. Some examples of these classes are listed as follows:

a. AG classes (Part 1 & 2) 5 GDI personnel attended
b. Ada Development class (UAH)
c. SMAP Project Management Workshop
d. TMIS PALS class

2.6.2 Approximately 10 Seminars Attended

There were several seminars which were attended by GDI personnel which were relevant to the activities of the Space Station. There were approximately 10 seminars which were attended. Some examples are listed below:

a. DMS Prototype Kit Seminar
b. Alsys Ada Seminar (2)
c. Gensym G2 Real-Time Development Environment
d. Mizar Real-Time Systems
e. Oasis Orientation Meeting
f. Loral Real-Time Systems
g. SMAP Seminar
h. NASA Workstation Seminar

3.0 HARDWARE PURCHASED

At the request of EB42, GDI purchased some hardware to enhance systems in the EB42 LAB. These items were needed to perform the necessary activities on the Space Station project.

3.1 MicroVax Hardware Upgrade

GDI purchased several components to update the MicroVax with a major upgrade. This was used to bring the MicroVax up to the needed operational level for the required work that was to be performed on it.

3.2 Apollo Disk Upgrade

In support of the SSE loads, it was made necessary to purchase a larger disk for the Apollo. The massive amount of software received from the SSE was to large to fit on the 77 MB disk with which the Apollo was configured. Therefore, GDI purchased a 155 MB Winchester disk to upgrade the system, so the incoming SSE software could be loaded onto the Apollo.

4.0 HARDWARE/SOFTWARE DEVELOPED

GDI's contract concentrated on two main areas of development over the life of the contract. These two areas consisted of the Application Generator and the SoftPanel Prototype display system. The Application Generator (AG) is a tool that Boeing selected to use to automate the software development effort. In conjunction with this selection, GDI was tasked with developing a control system on which the AG could be tested. The SoftPanel Prototype attempts to begin to address a number of Station issues including the complexity of the underlying system, the large number of critical parameters, reliance on multi-function displays, and on-board data processing and autonomy.

4.1 Hardware Developed

The only hardware developed specifically under this contract was the Aquarium system for the AG experiment.

4.1.1 Aquarium System

The Aquarium system was designed and built by GDI in support of the AG experiment. This hardware was built with the necessary instrumentation to allow a complete analysis of the development of control algorithms using the AG. The Aquarium system contained such items as pumps, thermocouples, lights, switches, etc. to provide a reasonable system to drive from the AG. The hardware components used in the Aquarium system are outlined completely in Section 3 of the "Aquarium Control System Requirements Document" (TR-2210/0005/001/00).

4.2 Software Developed

The software development effort under GDI's contract consisted of the AG control and the SoftPanel Prototype software. The AG generates Ada control software from a set of control diagrams. The SoftPanel Prototype code consisted of two stages of development using DataViews during the first stage and TAE+/XWindows during the second.

4.2.1 AG Control Diagrams and Ada Code for Aquarium System

The AG generates Ada software from the AG control diagrams. The generated software is downloaded to the AG controller to operate the Aquarium system hardware. During execution of the AG generated software, the user may specify set points, such as temperature or water level, at the AG workstation to exercise the Aquarium system hardware. Further discussion of the AG control software for the Aquarium system is discussed in Section 4 of the "Aquarium Control System Requirements Document" (TR-2210/0005/001/00).

4.2.1.1 Current Project Status

Current controls for Experiment 1 will increase the water temperature of Tank 1. The user specifies a temperature set point at the AG workstation. The AG controller compares the actual water temperature of Tank 1 to the temperature set point. If the actual temperature is below the temperature set point, the AG controller will turn the heater on and pump water in Tank 1 through the heater to obtain the temperature set point. The controls read a water level constant of three inches for Tank 1's water level and monitor high and low level alarm switches. The user can turn the Aquarium system light on and off at the AG workstation.

Ada software has been generated from the Experiment 1 control algorithms. The control algorithms have been successfully simulated. Experiment 1 controller software has been generated, downloaded, and interfaced with the Aquarium system hardware. The controller software will turn the Aquarium system light on or off, read and display water temperature of Tank 1, accept a user specified temperature set point, monitor high level and low level alarms, display activation/deactivation of Pump Heater, display if Heater Switch is on or off, and display water level constant for Tank 1.

Problems encountered when interfacing with the Aquarium system hardware are as follows:

1) The thermocouple is damaged and requires replacement ($35).

2) Upon startup, analog outputs to Pump Heater and Heater Switch are -10 volts. When the controller is started, the initial voltage should be zero but is remaining at -10 volts. The IOC file's initial values for the Pump Heater and Heater Switch are 0.0 which is either not being read by the controller or is not equivalent to 0 volts. When the Pump Heater is displayed as "activated" and when the Heater Switch is displayed as "on" the voltage reading should be +10 volts but is remaining at -10 volts. This problem requires further investigation.

4.2.1.2 Software Deliveries

A hard and soft copy of the software files that were generated for the Aquarium system are being delivered with this report. The 3.5" floppy disk containing the files will be delivered to EB42. A hard copy of all of the listed files is included as <u>Appendix B</u> of this report. The aforementioned files are as follows:

<u>Hardcopy files:</u>

Animation.Cfg - Defines interactive animation environment.

Expl.Pic - Interactive animation for Experiment 1.

<u>Simulation Files:</u>

Aqua_Sys.Ps - Displays System Build super blocks executed during simulation. Super blocks are System, Aqua_Sys, Controller, and Tank.

Expl.Pic - Displays interactive animation for Experiment 1 with labels on. The labels represent System Build input/output signals during simulation and are generated from the real time file developed under System Build.

Aqua_Sys.IOC - Displays System Build inputs and outputs for simulation. Signal types (monitor) and attributes are defined for each input and output. This file is generated by the Hardware Connection Editor (HWCE).

Aqua_Sys.Ada - AG generated Ada software for Aquarium system simulation.

Hardware Interface Files:

Controller.Ps - System Build control algorithm for the Aquarium system controller. The controller super block is executed when interfacing to the real hardware. The controller super block has been changed to output "actual temperature" and "water level constant" for display in the interactive animation while exercising the real hardware.

Exp1.Pic - Displays interactive animation for Experiment 1 with labels on. The labels represent System Build input/output signals when interfacing with the Aquarium system hardware and are generated from the real time file developed under System Build. (Note, the input signal for Tank 1 temperature is "actual temperature" when interfacing with hardware and is "temperature" when simulating the model.)

Aqua_Sys.IOC - Displays System Build inputs and outputs which interface with hardware and/or the Interactive Animator. Interactive Animator signal types are monitor. Signal types are OPTO_DA7 (Digital/Analog outputs) and OPTO_ODC5AQ (Digital outputs) for hardware inputs and OPTO_AD12 (Analog/Digital inputs) and OPTO_IDC5BQ (Digital inputs) for hardware outputs. This file is generated by the HWCE.

Aqua_Sys.Ada - AG generated Ada software for exercising real hardware.

Backup Disk:

Aquarium System project files on disk are:

Animation.CFG
Aqua_Sys.ADA
Aqua_Sys.DAT
Aqua_Sys.IOC
Aqua_Sys.PIC
Aqua_Sys.RTF
Exp1.PIC
Exp2.PIC
Exp3.PIC
Exp4.PIC
Exp5.PIC
Exp6.PIC
IODEF.DAT
Target_Config.CFG

To copy disk:

Execute the **makeproject** command at the VAX $ prompt in project directory. Makeproject generates Animation.CFG, Project.ALB, and IODEF.DAT files. Copy all files from disk to directory where makeproject command was executed. The copy command will overwrite the makeproject generated Animation.CFG and IODEF.DAT files. The Aquarium system project can then be loaded into RTMONIT.

4.2.1.3 Follow-up Work

If it can be arranged for work to continue on the AG and especially the Aquarium system, the following is a listing of some follow-up activities that would be useful to continue the project:

1) Replace thermocouple.

2) Resolve control problems encountered when interfacing Experiment 1 controls with Aquarium system hardware.

3) Add controls to Experiment 1 to decrease Tank 1 water temperature and read actual water level of Tank 1 as defined in the Aquarium Control System Requirements Document.

4) Develop, simulate, and interface to hardware control algorithms for Experiments 2, 3, 4, 5, and 6 as defined in the Aquarium Control System Requirements Document.

4.2.2 SoftPanel Prototype

Graphical User Interface (GUI) development for a complex system such as the Space Station Freedom presents a number of difficulties not commonly found in commercial direct manipulation user environments. Many of the software systems associated with the Station are data-driven rather than user-driven. Also, the Station includes a wide variety of concurrent applications involving highly specialized areas of engineering and other disciplines; as such, there are potentially unique requirements in the area of visual display technology, which may or may not be satisfied using standard direct manipulation techniques. Furthermore, commonality requirements mean systems must be usable among many different users in different work environments.

The SoftPanel Prototype is a GUI developed by MSFC and General Digital Industries (GDI) in support of the Space Station Freedom user interface definition. It attempts to address a number of Station issues. These issues include the complexity of the underlying system, the large number of critical parameters, reliance on multi-function displays, and on-board data processing and autonomy. The SoftPanel Prototype incorporates direct manipulation, graphical representation and controls, and a window display environment to focus on the problems associated with user search activities in a complex system. The following is a description of the two different approaches taken in order to explore these issues.

4.2.2.1 Graphical User Interface - DataViews Approach

The first SoftPanel Prototype was developed on the Sun 03/260 workstation under UNIX version 3.2. It was developed using the C programming language and relies on a combination of two graphical user interface development resources. These include the SunView window environment and DataViews version 6.0. SunView is used to provide window management, keyboard and mouse communications. DataViews provides graphical presentation and control elements.

DataViews consists of two utility packages, DV-Tools and DV-Draw. DV-Tools is a library of subroutines used to manipulate dynamic graphical objects. DV-Draw is a two-dimensional drawing editor used to create drawings with static and dynamic components. DV-Draw also provides a collection of predefined virtual instruments such as graphs, gauges, dials, etc. to be used for data representation and dynamic simulations. Together, DV-Tools and DV-Draw provide developers with the ability to create complete graphical displays.

Integration of DataViews with SunView was necessary because neither package lent itself completely to the implementation of the advanced features required for the SoftPanel Prototype. DataViews provides the resources necessary for implementation and presentation of graphical dynamic virtual instrumentation, but it is weak in its user interaction features and it lacks window management features. SunView provides strong capabilities in window management and keyboard and mouse communications, but its facilities for producing dynamic graphics are relatively primitive. Although bringing these resources together proved to be a non-trivial programming task, the outcome is a development framework that supports rapid and flexible implementation of sophisticated direct manipulation interface prototypes.

The first approach organizes the Station's functions into a classification scheme encompassing Station elements, systems, subsystems, and instruments. These components are classified into a hierarchical index. Each hierarchical level defines the interrelationship among the system, functional, and physical viewpoints of the components. This interrelationship links these viewpoints together and allows the user to traverse throughout a tree structure providing an effective search strategy. Windows and instruments are used in order to accomplish its monitoring and control functions. Windows are the virtual terminals used to contain and manage the display of instrumentation. Instruments serve as the virtual devices used by the operator to interact with monitor and control activities, for example graphs, dials, maps, schematics, knobs, and meters.

There were several lessons learned from this approach. One lesson was that the displays were too cluttered with system schematics for even an army of individuals to monitor. There needed to be more displays with less information displayed and even then, possibly, much of the information being displayed was not necessary. Another lesson learned was that the hierarchy index, while well organized, was too cumbersome in which to quickly maneuver. There were too many paths to follow and even though each path was labeled, the user had several methods of arriving at the same goal. Often times the user could become confused with how he got to the goal in the first place. One important lesson learned was that DataViews was not the GUI software to be used for this complicated task. The software was just too limited in its graphical qualities to be useful for this prototype which defeats one of the purposes of its development.

A complete report of this phase of the project was submitted on May 25, 1989 under the report number TR 2210/0001/002/00.

4.2.2.2 Graphical User Interface - TAE+ Approach

The second SoftPanel Prototype was developed on the Sun 03/260 workstation under UNIX version 4.0.3. It was developed using X-Windows 11 Revision 4.0 and Transportable Application Environment (TAE+) version 4.1.

TAE+ was developed by Goddard Space Flight Center for the purpose of rendering quick interactive application systems. The toolset employees windows, graphical objects, icons, a large palette of colors and fonts, and a selection of input devices for data retrieval. TAE+ runs on top of X-Windows 11 Revision 3.0 or 4.0 using the X-Window manager to control the windowing functions of the application. These two software packages work well together without any extra interfacing which was required for SunViews and DataViews. TAE+ allows the developer to rapidly develop displays, connect them for interaction, and generate the code. The code can be generated in one of four languages: Ada, C, Fortran, and TAE Command Language. The software package is very robust and lends itself well to developing displays.

The second development approach to the SoftPanel Prototype used the User Interface Requirements Document (SY45.1) as the standard for designing the displays. For this Prototype, it was decided not to dissect the entire Station into a set of interrelated displays, but model only one system. The Environmental Control and Life Support System (ECLSS) was chosen. At the same time MSFC and Boeing were working on another project, the Environmental Control and Life Support System Advanced Automation Project, which needed a GUI for its test purposes. By teaming with this project our design was further enhanced by having the opportunity to link to real-time data to run our displays.

The GUI design uses many windows and functions for displaying information and a mouse for quick maneuverability. The Prototype is designed to immediately switch to any part of the ECLSS system, ranging from an overview of the entire system to any component level, with a minimum of mouse clicks (usually only one or two). This method allows the user to spend more time in system analysis and less time in system navigation. Each screen is equipped with a "quit" button for quick exits, a "help" button for screen explanations, a "Tools" pull-down menu for access to additional system tools such as notepads, system schematics, and maintenance logs, and a window label for window identification. The main focus of each window is a functional diagram of a system with some data being displayed. There are many selectable items within the diagram which the user may click on to obtain more information about the instrumentations health. The side of the window contains a selection of strip charts which monitor data trends to observe severe data skews.

To date this part of the project is incomplete. There have been many hardware failures with the Sun workstation. Also, the ECLSS Testbed has undergone major redesign which greatly impacted the display design. Therefore, work progressed much slower than anticipated. One lesson learned from the TAE+ prototype is that the more simple displays seem to work better for understanding the current status of the system and allows the user to determine at a glance, if there are instrumentation problems.

More detailed information may be obtained about the ECLSS Advanced Automation Project and the GUI from BOEING's 1990 Annual Report and project video.

4.2.2.3 Follow-up Work

Much work is left to be done on the TAE+ Prototype. Most of the displays needed to be finished. The data link from the ECLSS Testbed to the Sun workstation still needs to be completed. Additional research on TAE+ as a GUI tool needs to be explored. Also, after the code is generated for the GUI application, the developer will have to include provisions for reading and displaying data.

Included with this report will be three 1/4" tapes containing the X-Windows 11 Revision 4.0, TAE+ version 4.1 including a directory with the SoftPanel Prototype files, and a copy of the DataViews SoftPanel Prototype.

5.0 LEVEL II SUPPORT

In support of the Space Station contract, GDI was required to interface and provide support to the Level II effort. This included three main tasks which are discussed in the following paragraphs.

5.1 Level II SIM Support

GDI provided a lead person to develop a "Simulation Architecture" document for splinter group 3 of the Level II Simulation Task Team.

5.1.1 Lead on SIM Task Team SIM Architecture group

The "Simulation Architecture" document that was developed outlined a feasible simulation architecture that could be applied across the program. The purpose of this was to ensure that the various simulations being developed by the different work packages could communicate/work together in a testing/integration facility if needed. This common simulation architecture across the program enables the program to perform end-to-end testing of the entire Space Station, if deemed necessary by Level II management. This common architecture mitigates some of the risk involved in integrating and performing such a task.

5.2 Standards

GDI was tasked to upgrade and develop standards for MSFC which would ensure commonality of design within MSFC. The two main tasks undertaken were a re-write of the SMADR document as well as roll-out of the MASM section into a document of its own. In addition, GDI participated in the analysis of the Spiral Model for a later revision of the SMADR.

5.2.1 SMADR Officially Signed-off on by NASA/MSFC

GDI received the task in 1988 to update the SMADR, MA-001-006-2H, dated January 1983. But in CY1990, MSFC decided to upgrade the document to make it applicable to all MSFC contractors and MSFC software development efforts, opposed to being applicable to only "EB" Laboratory software development efforts. The task included revising and updating Chapters 1, 2, 3 and the Data Requirements Appendix. In addition, the task included rolling out the Methods and Standards Section of the SMADR into a new document called the MSFC Methods and Standards Manual (MASM) (see paragraph 5.2.2 for a discussion of this task).

The final draft of the Software Management and Development Requirements (SMADR) document was delivered to MFSC's, EB-41 in Calendar year 1990. The MSFC Center Director, Jack Lee, signed the SMADR, MMI 8075.1, in January, 1991.

5.2.2 MASM Presently Under Review by NASA/MSFC

GDI was directed to roll-out the Methods and Standards Section of the SMADR and create a new document called the MSFC MASM. GDI delivered the first draft of the MASM to MFSC, EB-41 in December 1989. The NASA RID review of the MASM is still an open item as of February 15, 1991. The update effort included revising the existing methods and standards information to bring it in line with new technology, and to address new topic areas in the MASM. The MASM document now consists of over 100 pages of information. These additions, to the MASM, are as follows:

a. Software Development Methods by Phase
b. Programming Language Selection & Usage
c. Resource Margins
d. Documentation
e. User Interface

5.2.3 Spiral Model Development

GDI personnel reviewed the proposed Spiral Model Development updates to the SMADR, and provided comments to MSFC EB42 on the proposed changes. In addition, Randy Bounds, executive panel member, John Reynolds, Mike Faulkner and Jim Moon participated in the Spiral Model Workshop held at TRW's Huntsville, Alabama Facility, on 12/11/90.

5.3 Risk Management Assessment

As an add-on to the existing GDI contract, Randy Bounds managed a task in conjunction with TRW to perform a Risk Management Assessment of the Space Station. This included developing a plan to perform these actions, as well as, an assessment of the Space Station and of the present technology base available.

5.3.1 Risk Management Plan Developed

The first item of the add-on task was the development of a Risk Management Plan for the Space Station. This defined a process that would be required to ensure the identification of any risks to the program. This plan also identified a categorization of the risks and ways that the different types of risks could be mitigated from the program.

5.3.2 Risk Management Assessment of Space Station Software

After completing the Risk Management Plan, an assessment of the Space Station software was performed using the guidelines outlined in the plan (section 5.3.1). Approximately 95 risks were identified as a result of the Risk Management assessment. In conjunction with the risks identified, risk mitigation scenarios for alleviating the risk areas were also identified.

5.3.3 Assessment of Needed/Available Technology Base

The final part of the task was to identify the available technology base in contrast to what technology is needed. This included the identification of CASE tools and other standardized tools which would aid in standardizing the development effort. Use of industry standardized tools would identify a technology base which could be used to aid in the mitigation of the risks identified under the previous task.

APPENDIX A

WORKING PAPERS/TECHNICAL REPORTS LISTING

NASA Document Series
General Digital Industries (GDI)
Working Paper (WP)

GDI Control #	Section Name	Title	Date	Author
WP-2210/0001/001/00	Soft Panel Testbed WP	ECLSS WG Action Item	06/02/89	GDI
WP-2210/0001/002/00	Soft Panel Testbed WP	SUN 3/260 Workstation Problem Report	12/21/90	KM
WP-2210/0002/001/DR	Document Standards	SMADR Tailoring	08/03/88	AP
WP-2210/0002/003/00	Document Standards	Review of DRLI 93, 4.0, Versus SMAP 3.0	02/08/90	JDM
WP-2210/0002/004/00	Document Standards	Review of CR 88001020, Baseline of SSFP SSE Docume	07/31/90	JDM
WP-2210/0003/001/00	Software Engineering	Payload Software Engineering Management Process	10/11/89	JDM
WP-2210/0003/002/00	Software Engineering	Lynx OS Information	01/02/90	MAF
WP-2210/0003/003/00	Software Engineering	SSE/Lockheed Rational Computer Briefing	02/28/90	JDR
WP-2210/0003/004/00	Software Engineering	Notes Concerning I&V Process and C. Nola's Paper	03/20/90	GDI
WP-2210/0003/005/00	Software Engineering	Rational Computer Information Search Report	03/12/90	JDR
WP-2210/0004/001/00	Project Status Reports	June Status Report (2 Copies)	07/13/88	JDR
WP-2210/0004/002/00	Project Status Reports	July Status Report	08/09/88	JDR
WP-2210/0004/003/00	Project Status Reports	August/September Status Report	10/05/88	JDR
WP-2210/0004/004/00	Project Status Reports	October/November Status Reports	12/12/88	JDR
WP-2210/0004/005/00	Project Status Reports	December - February Status Reports	03/13/89	JDR
WP-2210/0004/006/00	Project Status Reports	March/April Status Reports	05/12/89	JDR
WP-2210/0004/007/00	Project Status Reports	Soft Panel Testbed Status Report for May 1989	06/01/89	JDR
WP-2210/0004/008/00	Project Status Reports	Soft Panel Testbed Status Report for June 1989	07/31/89	JDR
WP-2210/0004/009/00	Project Status Reports	July/August 1989 Status Report	09/05/89	JDR
WP-2210/0004/010/00	Project Status Report	Soft Panel Testbed Status Report for September 198	10/09/89	GDI
WP-2210/0004/011/00	Project Status Reports	Soft Panel Testbed Status Report for 10/89 - 11/89	12/07/89	JDR
WP-2210/0004/012/00	Project Status Report	Soft Panel Testbed Status Report for December 1989	01/02/90	JDR
WP-2210/0004/013/00	Project Status Report	Soft Panel Testbed Status Report for January 1990	02/01/90	JDR
WP-2210/0004/014/00	Project Status Report	Soft Panel Testbed Staus Report for February 1990	03/01/90	JDR
WP-2210/0004/015/00	Project Status Report	Soft Panel Testbed Status Report for March 1990	04/02/90	GDI
WP-2210/0004/016/00	Project Status Report	Soft Panel Testbed Status Report for April/May 90	06/06/90	GDI
WP-2210/0004/017/00	Project Status Reports	Soft Panel Testbed Status Report for June 1990	07/09/90	JDR
WP-2210/0004/018/00	Project Status Reports	Soft Panel Testbed Status Report fro Jully 1990	08/17/90	GDI
WP-2210/0004/019/00	Project Status Reports	Soft Panel Testbed Status Report for August 1990	09/07/90	JDR
WP-2210/0004/020/00	Project Status Reports	Soft Panel Testbed Status Report for September 90	10/03/90	JDR
WP-2210/0004/021/00	Project Status Reports	Soft Panel Testbec Status Report for October 1990	11/06/90	JDR
WP-2210/0004/022/00	Project Status Reports	Soft Panel Testbed Status Report for November 1990	12/05/90	GDI
WP-2210/0004/023/00	Project Status Reports	Soft Panel Testbed Status Report for December 1990	01/10/91	GDI
WP-2210/0004/024/00	Project Status Reports	Soft Panel Prototype Status Report January 1991	02/05/91	JDR
WP-2210/0005/001/00	Meeting Reports	GDI Report on Application Generator Demonstration	07/05/88	JDR
WP-2210/0005/002/00	Meeting Reports	Report on OSID Telecon	07/14/88	JDR
WP-2210/0005/003/00	Meeting Reports	Application Generator Assessment	07/29/88	JDR
WP-2210/0005/004/00	Meeting Reports	USE Telecon	02/15/89	AP
WP-2210/0005/005/00	Meeting Reports	USEWG Meeting	04/05/89	MRB
WP-2210/0005/006/00	Meeting Reports	USE OPS CUI Telecon	04/11/89	MRB
WP-2210/0005/007/00	Meeting Reports	UAH Avionics Distributed System Demo	05/25/89	MRB
WP-2210/0005/008/00	Meeting Reports	ECLSS Reliability Analysis Kickoff Meeting	06/27/89	WOW
WP-2210/0005/009/00	Meeting Reports	SSE PDR Coordination Telecon (7/21/89)	07/25/89	JDM
WP-2210/0005/010/00	Meeting Reports	SSE PDR Telecon (8/2-3/89)	08/04/89	JDM
WP-2210/0005/011/00	Meeting Reports	SSE Simulation Standards Telecon - 9/25/89	09/25/89	WOW
WP-2210/0005/012/00	Meeting Reports	Global Inventory Management System (IMS) Meeting	01/11/90	MAF
WP-2210/0005/013/01	Meeting Reports	SSE User's Working Group Discussion of Software St	01/25/90	MC
WP-2210/0005/014/00	Meeting Reports	SSE User's Working Group	04/17/90	YML
WP-2210/0006/001/00	Trip Reports	USE Working Group Meeting (University of Colorado)	07/08/88	AP

ORIGINAL PAGE IS OF POOR QUALITY

NASA Document Series
General Digital Industries (GDI)
Working Paper (WP)

GDI Control #	Section Name	Title	Date	Author
WP-2210/0006/002/00	Trip Reports	Goddard Space Flight Center (5/18 - 5/19)	05/23/89	AP
WP-2210/0006/003/00	Trip Reports	DMS WG Meeting at JSC May 23-25, 1989	06/07/89	MAF
WP-2210/0006/004/00	Trip Reports	DMS/DMA SDR/SRR - JSC Trip	07/14/89	MAF
WP-2210/0006/005/00	Trip Reports	USE WG Meeting (KSC) 8/1-3/89	08/09/89	AP
WP-2210/0006/006/00	Trip Reports	Trip Report for DRLI 93 CC3 and SSE PDR	08/23/89	JDM
WP-2210/0006/007/00	Trip Reports	DMS WG Meeting - JSC Trip Report (10/16-20/89)	10/24/89	MAF
WP-2210/0006/008/00	Trip Reports	DMS PDR & SIB DRR - JSC Trip Report	12/04/89	MAF
WP-2210/0006/009/00	Trip Reports	DRLI 59 Design Implementation Review Report	12/21/89	JDR
WP-2210/0006/010/00	Trip Reports	DMS Software PDR	02/13/90	MAF
WP-2210/0006/011/00	Trip Reports	SSFP UIL Spec. RID Disposition Board Meeting	03/06/90	AP
WP-2210/0006/012/00	Trip Reports	Workshop on NASA Workstation Terminology	03/13/90	AP
WP-2210/0007/001/00	DMS/SIB Reports	SIB - Time Tag & Run/Idle Designs	01/12/89	MAF
WP-2210/0007/002/01	DMS/SIB Reports	SIB - Time Inerface and Simulation Cycle Designs	01/20/89	MAF
WP-2210/0007/003/01	DMS/SIB Reports	DMS Kits Road Show Agenda	01/25/89	MAF
WP-2210/0007/004/01	DMS/SIB Reports	SIB - Local Bus Interface Design	01/25/89	MAF
WP-2210/0007/005/00	DMS/SIB Reports	SIB MDM/ORU Interface Design	02/02/89	MAF
WP-2210/0007/006/00	DMS/SIB Reports	SDDU Interface Design	02/08/89	MAF
WP-2210/0007/007/00	DMS/SIB Reports	SIB FDDI Interface Design	02/15/89	MAF
WP-2210/0007/008/00	DMS/SIB Reports	Data Recording	02/23/89	MAF
WP-2210/0007/009/00	DMS/SIB Reports	DMS Kits Road Show Summary	02/23/89	MAF
WP-2210/0007/010/00	DMS/SIB Reports	DMS Kits Road Show Trip Report	03/02/89	MAF
WP-2210/0007/011/00	DMS/SIB Reports	SIB Data Flow Diagrams	03/08/89	MAF
WP-2210/0007/012/00	DMS/SIB Reports	DMS/SIB Mode Team Update	03/17/89	MAF
WP-2210/0007/013/00	DMS/SIB Reports	SIB Diagnostics	03/23/89	MAF
WP-2210/0007/014/00	DMS/SIB Reports	SIB Time Interface Requirements	03/29/89	MAF
WP-2210/0007/015/00	DMS/SIB Reports	DMS Kits Telecon	04/04/89	MAF
WP-2210/0007/016/00	DMS/SIB Reports	SIB H/W & S/W Detailed Requirements Spec. (DRLI 76	04/14/89	MAF
WP-2210/0007/017/00	DMS/SIB Reports	DMS - MSIF Preliminary Requirements	04/11/89	MAF
WP-2210/0007/018/00	DMS/SIB Reports	SIB/SIM Concepts Update	04/12/89	MAF
WP-2210/0007/019/00	DMS/SIB Reports	SIB - Simulation and Modeling COTS Tools	04/19/89	MAF
WP-2210/0007/020/00	DMS/SIB Reports	Questions on the New Suggested SIB/SIPS Design	04/24/89	MAF
WP-2210/0007/021/00	DMS/SIB Reports	SIB Ground Isolation Requirements Overview	04/26/89	MAF
WP-2210/0007/022/00	DMS/SIB Reports	SIB - SSE Support to SSFP IT&V	05/04/89	MAF
WP-2210/0007/023/00	DMS/SIB Reports	DMS OS/Ada RTE Software Requirements Specification	05/04/89	JM
WP-2210/0007/024/00	DMS/SIB Reports	DMS Kits CEI Specification Review Update	05/10/89	MAF
WP-2210/0007/025/00	DMS/SIB Reports	Proposed DMS Release Contents	05/19/89	MAF
WP-2210/0007/026/00	DMS/SIB Reports	SIB - SSE Simulation Operations Concept	05/19/89	MAF
WP-2210/0007/027/00	DMS/SIB Reports	Early Support for DMS Users	06/08/89	MAF
WP-2210/0007/028/00	DMS/SIB Reports	Simulation Architecture Selection Study Overview	06/08/89	MAF
WP-2210/0007/029/00	DMS/SIB Reports	Reconfigurability of Kits	06/15/89	MAF
WP-2210/0007/030/00	DMS/SIB Reports	Ada Compiler Evaluation	06/15/89	MAF
WP-2210/0007/031/00	DMS/SIB Reports	Boeing DMS Baseline Design Review (BDR)	06/19/89	MAF
WP-2210/0007/032/00	DMS/SIB Reports	SIB Data Transfer Concept	06/22/89	MAF
WP-2210/0007/033/00	DMS/SIB Reports	General Comments on Current DMS Events	06/28/89	MAF
WP-2210/0007/034/00	DMS/SIB Reports	SIPS Simulation Control Design Concept	06/28/89	MAF
WP-2210/0007/035/00	DMS/SIB Reports	DMS Performance WG Telecon	07/06/89	MAF
WP-2210/0007/036/00	DMS/SIB Reports	SIB Forum - SIM Architecture Selection Study Resul	07/24/89	MAF
WP-2210/0007/037/00	DMS/SIB Reports	DMS Kits - SIM Architecture Selection Study Status	07/25/89	MAF
WP-2210/0007/038/00	DMS/SIB Reports	SIB Forum Telecon	07/26/89	MAF

ORIGINAL PAGE IS
OF POOR QUALITY

NASA Document Series
General Digital Industries (GDI)
Working Paper (WP)

GDI Control #	Section Name	Title	Date	Author
WP-2210/0007/039/00	DMS/SIB Reports	DMS Kits Utilization	08/14/89	MAF
WP-2210/0007/040/00	DMS/SIB Reports	SIB Forum - SIB Multiprocessor Study Overview	08/14/89	MAF
WP-2210/0007/041/00	DMS/SIB Reports	DMS Kits Telecon - Scrub 89 Discussion	08/15/89	MAF
WP-2210/0007/042/00	DMS/SIB Reports	DMS Kits Telecon - Round-table Discussion	08/31/89	MAF
WP-2210/0007/043/00	DMS/SIB Reports	SIB - SSFP IT&V Environment - Issues/SSE Recommend	09/13/89	MAF
WP-2210/0007/044/00	DMS/SIB Reports	SIB Development Activities	09/20/89	MAF
WP-2210/0007/045/00	DMS/SIB Reports	VAX ELN Real-Time Operating System	09/27/89	MAF
WP-2210/0007/046/00	DMS/SIB Reports	DMS Kits Teleconference	10/03/89	MB
WP-2210/0007/047/00	DMS/SIB Reports	SIB - DRLI 76 RID Disposition Open Issues	10/11/89	MAF
WP-2210/0007/048/00	DMS/SIB Reports	DMS Status Meeting	10/13/89	MAF
WP-2210/0007/049/00	DMS/SIB Reports	Remote Instrumentation Concentrator (RIC) Telecon	10/31/89	MAF
WP-2210/0007/050/00	DMS/SIB Reports	DMS Kit Component Allocations	10/31/89	MAF
WP-2210/0007/051/00	DMS/SIB Reports	DMS Performance Teleconference	11/02/89	MAF
WP-2210/0007/052/00	DMS/SIB Reports	DMS Status Meeting	11/08/89	MAF
WP-2210/0007/053/00	DMS/SIB Reports	DMS - Level II Status	11/14/89	MAF
WP-2210/0007/054/00	DMS/SIB Reports	SIB Detailed Requirements Review (DRR) Overview	11/17/89	MAF
WP-2210/0007/055/00	DMS/SIB Reports	Internal Audio & Video (IT&V) Meeting	11/17/89	MAF
WP-2210/0007/056/00	DMS/SIB Reports	DMS Services Provided	11/21/89	MAF
WP-2210/0007/057/00	DMS/SIB Reports	SIB - Summary and Overview of Prototype ITVE Demon	11/27/89	MAF
WP-2210/0007/058/00	DMS/SIB Reports	DMS Kits Survey Form	12/12/89	MAF
WP-2210/0007/059/00	DMS/SIB Reports	DMS Status Meeting	01/10/90	MAF
WP-2210/0007/060/00	DMS/SIB Reports	DMS Status Presentation	01/19/90	MAF
WP-2210/0007/061/00	DMS/SIB Reports	DMS Performance Telecon	01/18/90	MAF
WP-2210/0007/062/00	DMS/SIB Reports	DMS Performance Telecon - MDSSC User Survey Status	02/02/90	MAF
WP-2210/0007/063/00	DMS/SIB Reports	DMS Kits Telecon	02/12/90	MAF
WP-2210/0007/064/00	DMS/SIB Reports	DMS Kits - DMS PDR2 Issues and MDM FEU Overview	03/13/90	MAF
WP-2210/0007/065/00	DMS/SIB Reports	DMS Performance Videocon	03/23/90	MAF
WP-2210/0007/066/00	DMS/SIB Reports	DMS Kits Telecon	03/28/90	MAF
WP-2210/0007/067/00	DMS/SIB Reports	DMS Kits Telecon - DMS PDR #3	05/23/90	MAF
WP-2210/0007/068/00	DMS/SIB Reports	DMS Command and Control Meeting	05/30/90	MAF
WP-2210/0007/069/00	DMS/SIB Reports	DMS -CR to Incorporate DMS ACD Part 4	06/05/90	MAF
WP-2210/0007/070/00	DMS/SIB Reports	DMS Command & Control Telecon	06/06/90	MAF
WP-2210/0007/071/00	DMS/SIB Reports	DMS Status Meeting	06/29/90	MAF
WP-2210/0007/072/00	DMS/SIB Reports	DMS Kits telecon - GSI interface	07/05/90	MAF
WP-2210/0007/073/00	DMS/SIB Reports	GSE Interface Meeting - IEEE 488 versus Ethernet	07/11/90	MAF
WP-2210/0007/074/00	DMS/SIB Reports	DMS Performance Videocon	07/12/90	MAF
WP-2210/0007/075/00	DMS/SIB Reports	MDM FEU Overview	08/24/90	MAF
WP-2210/0007/076/00	DMS/SIB Reports	Comments on Alternate DMS Feasibility Study	08/30/90	MAF
WP-2210/0007/077/00	DMS/SIB Reports	DMS Kits WG Meeting - Multi-FEU MDM	09/19/90	MAF
WP-2210/0007/078/00	DMS/SIB Reports	DMS Kits - SIB CDR #1	11/07/90	MAF
WP-2210/0007/079/00	DMS/SIB Reports	DMS Kits Telecon	12/05/90	MAF
WP-2210/0008/001/00	MSIF Reports	MSIF Requirements Team Telecon Minutes	02/16/89	JDR
WP-2210/0008/002/00	MSIF Reports	MSIF Functional Requirements Team Telecon 3/14/89	03/15/89	WOW
WP-2210/0008/003/00	MSIF Reports	MSIF Team Lead Telecon 3/10/89	03/15/89	WOW
WP-2210/0008/004/00	MSIF Reports	Trip Report - MSIF SIM Task Team Activation Meetin	03/06/89	WOW
WP-2210/0008/005/00	MSIF Reports	MSIF Management Team Telecon 3/3/89	03/16/89	WOW
WP-2210/0008/006/00	MSIF Reports	MSIF SIM Task Team Telecon 3/17/89	03/22/89	WOW
WP-2210/0008/007/00	MSIF Reports	MSIF Team Lead Telecon 3/24/89	03/24/89	WOW
WP-2210/0008/008/00	MSIF Reports	Trip Report - MSIF SIM Services Tiger Team Meeting	04/17/89	WOW

ORIGINAL PAGE IS
OF POOR QUALITY

NASA Document Series
General Digital Industries (GDI)
Working Paper (WP)

GDI Control #	Section Name	Title	Date	Author
WP-2210/0008/009/00	MSIF Reports	Trip Report - MSIF SIM Task Team Prime Writers Mee	04/19/89	WOW
WP-2210/0008/010/00	MSIF Reports	MSIF Joint Team Telecon 4/11/89	04/19/89	WOW
WP-2210/0008/011/00	MSIF Reports	ICHSTF Lessons Learned for MSIF	04/05/89	MAF
WP-2210/0008/012/00	MSIF Reports	MSIF Requirements Team Videocon - Status Report	04/07/89	MAF
WP-2210/0009/003/00	Weekly Status Meeting	Weekly NASA/MSFC Status Meeting Minutes	02/17/89	JDR
WP-2210/0009/004/00	Weekly Status Meeting	Weekly NASA/MSFC Status Meeting Minutes 3/1/89	03/01/89	JDR
WP-2210/0009/005/00	Weekly Status Meeting	Weekly NASA/MSFC Status Meeting Minutes 4/5/89	04/10/89	MSF
WP-2210/0010/001/00	S/W Managers Meeting	Software Managers Videocon (in notebook)	02/02/89	WOW
WP-2210/0010/002/00	Software Managers	October 5, 1989 SSE Software Managers Meeting Minu	10/08/89	JDR
WP-2210/0011/001/00	SSE T&O Reports	Weekly T&O Telecon Minutes	03/02/89	JDR
WP-2210/0011/002/00	SSE T&O Reports	Weekly T&O Telecon Minutes	03/30/89	MAF
WP-2210/0011/003/00	SSE T&O Reports	SSE CCB Telecon Minutes	03/30/89	MAF
WP-2210/0011/004/00	SSE T&O Reports	Weekly T&O Telecon Minutes	04/06/89	MAF
WP-2210/0011/005/00	SSE T&O Reports	Weekly T&O Telecon Minutes	04/12/89	JDR
WP-2210/0011/006/00	SSE T&O Reports	Weekly T&O Telecon Minutes	04/27/89	JDR
WP-2210/0011/007/00	SSE T&O Reports	Weekly T&O Teleconference Minutes	05/05/89	JDR
WP-2210/0011/008/00	SSE T&O Reports	Weekly T&O Telecon Minutes	05/19/89	MAF
WP-2210/0011/009/00	SSE T&O Reports	June 6, 1989 SSE T&O Telecon Minutes	06/15/89	JDR
WP-2210/0011/010/00	SSE T&O Reports	June 15, 1989 SSE T&O Telecon Minutes	06/15/89	JDR
WP-2210/0011/011/00	SSE T&O Reports	June 29, 1989 SSE T&O Telecon Minutes	07/06/89	JDR
WP-2210/0012/001/00	OMA/Level II Reports	Element OMAs for WP01	03/02/89	GDI
WP-2210/0012/002/00	OMA/Level II Reports	Element OMAs for WP02	03/02/89	GDI
WP-2210/0012/003/00	OMA/Level II Reports	RIDs on Level II Software Policies Document	03/06/89	GDI
WP-2210/0012/004/00	OMA/Level II Reports	OMS Prime Item Development Specification (DR SY-06	04/06/89	GDI
WP-2210/0012/005/00	OMA/Level II Reports	OMA Interface Requirements Specification (DR SY-23	04/06/89	GDI
WP-2210/0012/006/00	OMA/Level II Reports	OMA Software Requirements Specification (DR SY-34)	04/05/89	GDI
WP-2210/0012/007/00	OMA/Level II Reports	OMS Software Test Plan (DR SY-35)	04/05/89	GDI
WP-2210/0012/008/00	OMA/Level II Reports	Verification and Validation Plan (DR SY-42)	04/04/89	GDI
WP-2210/0013/001/00	Application Generator	Comments on Boeings AG Ada SLOC Estimation	03/23/89	WOW
WP-2210/0013/002/00	Application Generator	AG Evaluation Task	03/27/89	MRB
WP-2210/0013/003/00	Application Generator	AG Meeting	04/28/89	MRB
WP-2210/0013/004/00	Application Generator	Application Generator Evaluation Criteria	05/13/89	MRB
WP-2210/0013/005/00	Application Generator	Comments on D683-10138-1, Draft 3, Sec. 5.4.1 AG S	07/20/89	JC
WP-2210/0013/006/00	Application Generator	AG Evaluation Task Proposal	07/25/89	JC
WP-2210/0013/007/00	Application Generator	Possible Application Generator Applications	08/07/89	GDI
WP-2210/0013/008/00	Application Generator	Recommendation for Application Generator Evaluatio	11/20/89	GDI
WP-2210/0013/009/00	Application Generator	Aquarium Control System Hardware Review	02/05/90	PT
WP-2210/0013/010/00	Application Generator	Application Generator Tutorial Review	02/21/90	MC
WP-2210/0013/011/00	Application Generator	Request to House Application Generator at GDI	03/12/90	PT
WP-2210/0013/012/00	Application Generator	Application Telecon for Users Group	03/23/90	MC
WP-2210/0013/013/00	Application Generator	Application Generator Teleconference	04/12/90	MC
WP-2210/0013/014/00	Application Generator	Application Generator Teleconference	04/26/90	MC
WP-2210/0013/015/00	Application Generator	Aquarium Control System Hardware Cost	06/20/90	PT
WP-2210/0013/016/00	Application Generator	AG User's Working Group	08/29/90	TML
WP-2210/0013/017/00	Application Generator	AG User's Working Group Telecon	10/09/90	TML
WP-2210/0013/018/00	Application Generator	AG User's Working Group Telecon	10/31/90	TML
WP-2210/0013/019/00	Application Generator	AG User's Working Group Telecon	11/27/90	TML
WP-2210/0013/020/00	Application Generator	Aquarium System Progress Report	01/07/91	TML
WP-2210/0013/021/00	Application Generator	Application Generator User's Working Group Telecon	01/23/91	TML

ORIGINAL PAGE IS
OF POOR QUALITY

NASA Document Series
General Digital Industries (GDI)
Working Paper (WP)

GDI Control #	Section Name	Title	Date	Author
WP-2210/0013/022/00	Application Generator	Aquarium System Progress Report	02/13/91	TML
WP-2210/0014/001/00	USE	USE MSFC ad hoc Meeting 4/24/89	04/24/89	AF
WP-2210/0014/002/00	USE	USE Telecon 4/25/89	04/25/89	AP
WP-2210/0014/003/00	USE	USE Telecon Minutes	05/01/89	MRB
WP-2210/0014/004/00	USE Reports	DMS USE SRS	05/11/89	AP
WP-2210/0014/005/00	USE Reports	USE WG Telecon 5/31/89	06/01/89	AP
WP-2210/0014/006/00	USE Reports	Displays Development Meeting	06/06/89	MRB
WP-2210/0014/007/00	USE Reports	Displays Development Meeting II	06/20/89	GDI
WP-2210/0014/008/00	USE Reports	DMA Interface Definition Process Meeting	06/22/89	GDI
WP-2210/0014/009/00	USE Reports	USE Telecon Meeting Notes	06/28/89	MRB
WP-2210/0014/010/00	USE Reports	USE Telecon - HCIG UIL and X Windows CR	10/10/89	MB
WP-2210/0015/001/00	I&V Reports	Boeing Simulation Questionnaire	06/06/89	WOW
WP-2210/0015/002/00	I&V Reports	SSFP I&V Flow Team Videocon - May 25, 1989	06/13/89	WOW
WP-2210/0015/003/00	I&V Reports	IV&T Team Videocon	06/28/89	WOW
WP-2210/0015/004/00	I&V Reports	IV&T Team Telecon 6/29/89	07/05/89	WOW
WP-2210/0015/005/00	I&V Reports	IV&T Team Meeting (7/20-21/89)	08/01/89	WOW
WP-2210/0016/001/00	SSE CCB Reports	June 8, 1989 SSE CCB Telecon Minutes	06/15/89	JDR
WP-2210/0016/002/00	SSE CCB Reports	09/14/89 SSE CCB Telecon Minutes	09/14/89	JDM
WP-2210/0016/003/00	SSE CCB Reports	October 5, 1989 SSE Level 3 CCB Meeting Minutes	10/08/89	JDR
WP-2210/0016/004/00	SSE CCB Reports	SSE CCB November 21, 1989	11/29/89	JDM
WP-2210/0016/005/00	SSE CCB Reports	SSE CCB December 19, 1989	12/22/89	JDM
WP-2210/0016/006/00	SSE CCB Reports	SSE Configuration Control Board (CCB) 05/24/90	05/24/90	JDM
WP-2210/0017/001/00	Payload Reports	PDRD to Payload Comparison	12/08/89	JDR
WP-2210/0018/001/00	Simulation Reports	SSFP Simulation Task Team Activation Meeting	01/04/90	WOW
WP-2210/0018/002/00	Simulation Reports	SSFP Simulation Team Splinter Group 1&2 Meeting	02/28/90	WOW
WP-2210/0018/003/00	ITVE/SIM Reports	ITVE PDR Summary Presentation	08/16/90	MAF
WP-2210/0018/004/00	ITVE/SIM Reports	ITVE PDR RID Review Trip Report	09/20/90	MAF
WP-2210/0018/005/00	ITVE/SIM Reports	Integrated Detailed Design Review (IDDR) #1	11/19/90	MAF
WP-2210/0019/001/00	CM Reports	SSFP Configuration Management (CM) Task Team Meeti	02/14/90	JDM
WP-2210/0019/002/00	CM Reports	Configuration Management Team Meeting	02/22/90	JDR
WP-2210/0020/001/00	SSEUWG	SSEUWG Meeting	02/16/90	JDM
WP-2210/0021/001/00	SEWG	SEWG telecon	03/16/90	MAF
WP-2210/0021/002/00	SEWG	Systems Environment Working Group (SEWG) Telecon	05/29/90	MAF
WP-2210/0021/003/00	SEWG	SEWG Telecon	06/07/90	MAF
WP-2210/0021/004/00	SEWG	SEWG Telecon	06/29/90	MAF
WP-2210/0021/005/00	SEWG	Systems Environment Working Group (SEWG)	10/08/90	MAF
WP-2210/0021/006/00	SEWG	Systems Environment Working Group (SEWG)	12/13/90	MAF
WP-2210/0022/001/00	SSFP DOC Library	GDI SSFP Documentation Library	12/21/90	JDM
WP-2210/0022/002/00	SSFP DOC Library	GDI SSFP Documentation Library	01/31/91	JDM

ORIGINAL PAGE IS
OF POOR QUALITY

NASA Document Series
General Digital Industries (GDI)
Technical Reports (TR)

GDI Control #	Section Name	Title	Date	Author
TR-2210/0001/001/00	Soft Panel Testbed	Soft Panel Prototype User's Manual	05/19/89	GDI
TR-2210/0001/002/00	Soft Panel Testbed	Soft Panel Prototype Technical Overview and Assess	05/26/89	GDI
TR-2210/0001/003/00	Soft Panel Testbed	Demonstration	05/10/89	GDI
TR-2210/0002/001/01	Document Standards	SMAP Information System Life-Cycle and Documentati	05/17/89	JM
TR-2210/0002/002/01	Document Standards	SMAP Version 4.3 Impact SS Software Management Pla	04/19/89	JM
TR-2210/0002/003/01	Document Standards	SSE Security Issues	06/01/89	JRM
TR-2210/0002/004/01	Document Standards TR	SSE Decnet Requirements	06/01/89	JRM
TR-2210/0002/005/00	Document Standards TR	Information Systems Life-Cycle and Documentation S	06/30/89	JRM
TR-2210/0002/006/00	Document Standards	Software Management and Development Requirements D	07/24/89	JDM
TR-2210/0002/007/00	Document Standards	Review of MSFC SMADR Update	10/31/90	JDM
TR-2210/0003/001/00	Software Engineering TR	Independent review of C3ATS	01/03/89	WOW
TR-2210/0003/002/00	Software Engineering	Comments on User Interface Language Specification	03/14/89	AP
TR-2210/0003/003/00	Software Engineering	Comments on IBM's Preliminary Req. Spec. for DMS U	03/17/89	AP
TR-2210/0003/004/00	Software Engineering	Shuttle To Station Software Comparison	05/23/90	MAF
TR-2210/0003/004/01	Software Engineering TR	Shuttle To Station Software Comparison (Revision)	06/29/90	MAF
TR-2210/0004/001/01	Reviews Reports	DRLI 94 Review (47 RIDs attached)	01/18/89	MAF
TR-2210/0004/002/01	Reviews Reports	SSE System Status Report (DRLI 5) (Notes & Comment	01/30/89	MAF
TR-2210/0004/003/00	Reviews Reports	RIDs on MSIF Concept Document	03/28/89	MAF
TR-2210/0004/004/00	Reviews Reports	System Management Plan (DRLI 01)	04/06/89	MAF
TR-2210/0004/005/00	Reviews Reports	SSE System Master Schedule (DRLI 97)	04/28/89	MAF
TR-2210/0004/006/00	Reviews Reports	SSE Requirements Specification (DRLI 72)	04/28/89	MAF
TR-2210/0004/007/00	Review Reports	WP01 SWPRR Review (RIDs/Comments)	05/11/89	MAF
TR-2210/0004/008/00	Review Reports	SSE System User Support Plan DRLI 12 RIDs	06/01/89	JDR
TR-2210/0004/009/00	Review Reports	SSE Software Product Specification OI 3.2 RIDs	06/15/89	MAF
TR-2210/0004/010/00	Review Reports	SSE System Documentation Standards DRLI 93 RIDs	07/11/89	JRM
TR-2210/0004/011/00	Review Reports	DMS SDR RIDs	07/19/89	MAF
TR-2210/0004/012/00	Review Reports	DMS/OMS SRR (RIDs)	07/21/89	GDI
TR-2210/0004/013/00	Review Reports	SSE System Element Design Document (DRLI 9) RIDs	07/25/89	JDR
TR-2210/0004/014/00	Review Reports	SSE System ICD (DRLI 21)	08/04/89	JDM
TR-2210/0004/015/00	Review Reports	SSE PDR (DRLI 18,21,59)	08/10/89	GDI
TR-2210/0004/016/00	Review Reports	Comments on SSFP Level II Software Policies Docume	08/11/89	GDI
TR-2210/0004/017/00	Review Reports	SIB Detailed Requirements Spec. (DRLI 76)	08/29/89	MAF
TR-2210/0004/018/00	Review Reports	Manned Base Reconfiguration Process FCD	10/25/89	MAF
TR-2210/0004/019/00	Review Reports	Review of DRLI 93, Version 4, Dated: October 9, 19	11/07/89	JDM
TR-2210/0004/020/00	Review Reports	SSE Software Product Specification (DRLI 94) OI4.0	01/16/90	JDR
TR-2210/0004/021/00	Review Reports	DRLI 59 Analysis	01/19/90	JDR
TR-2210/0004/022/00	Review Reports	SSE DIR DRLI 59/DRLI 72 Treaceability Analysis	02/08/90	JDR
TR-2210/0004/023/00	Review Reports	Software Review Board (SRB) Meeting	02/14/90	JDM
TR-2210/0004/024/00	Review Reports	WP01 SRS for the Airlock CSCI Review	04/02/90	TML
TR-2210/0004/025/00	Review Reports	SSFP Software Policies and Information System Stan	04/06/90	JDR
TR-2210/0004/025/00	Review Reports	DMS PDR3 Summary Reports	05/01/90	MAF
TR-2210/0004/027/00	Review Reports	Comments on DMS Development Plan (Draft 4)	06/28/90	MAF
TR-2210/0004/028/00	Review Reports	Comments/Impacts to CR BJ020388 on DMS ACD Part 4	07/10/90	MAF
TR-2210/0004/026/01	Review Reports	Comments/Impacts to CR BJ020388A on DMS ACD Part 4	07/13/90	MAF
TR-2210/0004/029/00	Review Reports	SSFP Software Life-Cycle Standards RIDs	07/18/90	JDM
TR-2210/0004/030/00	Review Reports	Comparison of DoD-STD-2167A, DRLI 93, SMAP 3.0, an	07/24/90	LDM
TR-2210/0004/031/00	Review Reports	DMS ACD Comments	10/04/90	MAF
TR-2210/0004/032/00	Review Reports	Comments on CR000839, Dated October 8, 1990	10/18/90	JDM
TR-2210/0004/033/00	Review Reports	Level A - Part II [DMS] Rewrite - Comments	12/14/90	MAF

ORIGINAL PAGE IS
OF POOR QUALITY

NASA Document Series
General Digital Industries (GDI)
Technical Reports (TR)

GDI Control #	Section Name	Title	Date	Author
TR-2210/0004/034/00	Review Reports	Firmware Analysis Presentation Comments	01/15/91	MAF
TR-2210/0005/001/00	Application Generator	Aquarium Control System Requirements Document	12/07/90	TML
TR-2210/0006/001/01	Product Reviews	MIZAR Seminar - Systems For Real-Time Applications	01/20/89	MAF
TR-2210/0006/002/00	Product Review Reports	Statemate from i-Logix	09/08/89	MAF
TR-2210/0006/003/00	Product Review Reports	Interleaf Desktop Publishing Tool Evaluation	01/31/91	TML

ORIGINAL PAGE IS
OF POOR QUALITY

NASA Document Series
General Digital Industries (GDI)
Software (SW)

GDI Control #	Section Name	Title	Date	Author
SW-2210/0001/001/00	Soft Panel Source Code	Soft Panel Prototype System	05/26/89	GDI

ORIGINAL PAGE IS
OF POOR QUALITY

APPENDIX B

APPLICATION GENERATOR DELIVERABLE FILES LISTING

AQUARIUM SYSTEM FINAL REPORT

Turnover Materials:

Hardcopy files:

Animation.Cfg - Defines interactive animation environment.

Expl.Pic - Interactive animation for Experiment 1.

Simulation Files:

Aqua_Sys.Ps - Displays System Build super blocks executed during simulation. Super blocks are System, Aqua_Sys, Controller, and Tank.

Expl.Pic - Displays interactive animation for Experiment 1 with labels on. The labels represent System Build input/output signals during simulation and are generated from the real time file developed under System Build.

Aqua_Sys.IOC - Displays System Build inputs and outputs for simulation. Signal types (monitor) and attributes are defined for each input and output. This file is generated by the Hardware Connection Editor (HWCE).

Aqua_Sys.Ada - AG generated Ada software for Aquarium System simulation.

Hardware Interface Files:

Controller.Ps - System Build control algorithm for the Aquarium System controller. The controller super block is executed when interfacing to the real hardware. The controller super block has been changed to output "actual temperature" and "water level constant" for display in the interactive animation while exercising the real hardware.

Expl.Pic - Displays interactive animation for Experiment 1 with labels on. The labels represent System Build input/output signals when interfacing with the Aquarium System hardware and are generated from the real time file developed under System Build. (Note, the input signal for Tank 1 temperature is "actual temperature" when interfacing with hardware and is "temperature" when simulating the model.)

Aqua_Sys.IOC - Displays System Build inputs and outputs which interface with hardware and/or the Interactive Animator. Interactive Animator signal types are monitor. Signal types are OPTO_DA7 (Digital/Analog outputs) and OPTO_ODC5AQ (Digital outputs) for hardware inputs and OPTO_AD12 (Analog/Digital inputs) and OPTO_IDC5BQ (Digital inputs) for hardware outputs. This file is generated by the HWCE.

Aqua_Sys.Ada - AG generated Ada software for exercising real hardware.

Backup Disk:

Aquarium System project files on disk are:

```
Animation.CFG
Aqua_Sys.ADA
Aqua_Sys.DAT
Aqua_Sys.IOC
Aqua_Sys.PIC
Aqua_Sys.RTF
Exp1.PIC
Exp2.PIC
Exp3.PIC
Exp4.PIC
Exp5.PIC
Exp6.PIC
IODEF.DAT
Target_Config.CFG
```

To copy disk:

Execute the **makeproject** command at the VAX $ prompt in project directory. Makeproject generates Animation.CFG, Project.ALB, and IODEF.DAT files. Copy all files from disk to directory where makeproject command was executed. The copy command will overwrite the makeproject generated Animation.CFG and IODEF.DAT files. The Aquarium System project can then be loaded into RTMONIT.

Final Project Status Report:

Current controls for Experiment 1 will increase the water temperature of Tank 1. The user specifies a temperature set point at the AG workstation. The AG controller compares the actual water temperature of Tank 1 to the temperature set point. If the actual temperature is below the temperature set point, the AG controller will turn the heater on and pump water in Tank 1 through the heater to obtain the temperature set point. The controls read a water level constant of three inches for Tank 1's water level and monitor high and low level alarm switches. The user can turn the Aquarium System light on and off at the AG workstation.

Ada software has been generated from the Experiment 1 control algorithms. The control algorithms have been successfully simulated. Experiment 1 controller software has been generated, downloaded, and interfaced with the Aquarium System hardware. The controller software will turn the Aquarium system light on or off, read and display water temperature of Tank 1, accept a user specified temperature set point, monitor high level and low level alarms, display activation/deactivation of Pump Heater, display if Heater Switch is on or off, and display water level constant for Tank 1.

Problems encountered when interfacing with the Aquarium System hardware are as follows:

1) Thermocouple is damaged.
2) Upon startup, analog outputs to Pump Heater and Heater Switch are -10 volts. When the controller is started, initial voltage should be zero but is remaining at -10 volts. The IOC file's initial values for the Pump Heater and Heater Switch are 0.0 which is either not being read by the controller or is not equivalent to 0 volts. When the Pump Heater is displayed as "activated" and when the Heater Switch is displayed as "on" the voltage reading should be +10 volts but is remaining at -10 volts.

Follow-up Work:

1) Replace thermocouple.

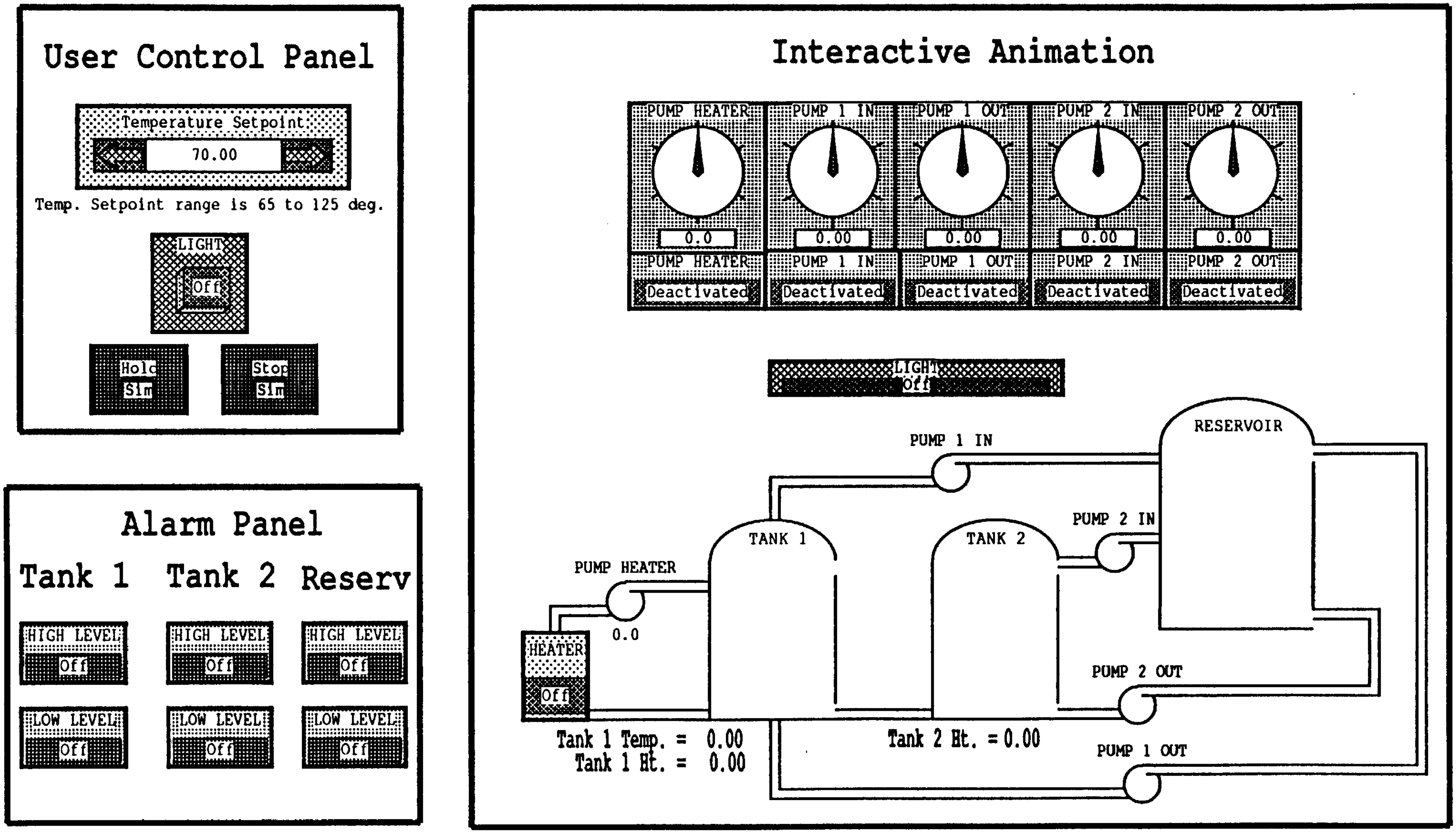
2) Resolve control problems encountered when interfacing Experiment 1 controls with Aquarium System hardware.

3) Add controls to Experiment 1 to decrease Tank 1 water temperature and read actual water level of Tank 1 as defined in the Aquarium Control System Requirements Document.

4) Develop, simulate, and interface to hardware control algorithms for Experiments 2, 3, 4, 5, and 6 as defined in the Aquarium Control System Requirements Document.

```
WS_DRAW CONFIGURATION FILE (ANIMATION.CFG) VERSION 2.0

*************************************************************
* The first line of the config file must be:                *
*   WS_DRAW CONFIGURATION FILE (ANIMATION.CFG) VERSION X    *
* All meaningful lines must start with the key words,       *
* followed by the name of the file in single quotes.        *
* Blank and comment lines are allowed.                      *
*                                                           *
* OTHER KEYWORDS INCLUDE THE FOLLOWING DEFAULTS:            *
*                                                           *
*      ICON_SOURCE_FILE    ==> 'ANIMATION:ICON.SRC'         *
*                                                           *
*      ICON_DATA_FILE      ==> 'ANIMATION:ICON.SOG'         *
*                                                           *
*      BUILD_CONTROL_PANEL ==> 'ANIMATION:CONTROL.SOG'      *
*************************************************************
C -- The following is the first animation picture to be loaded
BUILD_LOAD_PICTURE: 'AQUA_SYS.PIC'

C -- The following is a Matrixx FSAVE file used by the IA Animate Command
SIMULATION_DATA_FILE: 'AQUA_SYS.SIM'

C -- The following is the System Build Real-Time (.RTF) file
SYSTEM_BUILD_RTF_FILE: 'AQUA_SYS.RTF'

C -- The following is the code generation output (.Ada) file
CODE_GENERATION_OUTPUT_FILE: 'AQUA_SYS.ADA'

C -- The following is the Ada library to compile source into
ADA_LIBRARY: 'AQUA_SYS.ALB'

C -- Multiple of defined frequency at which to run the scheduler
C -- (To be prompted for the factor at run-time, specify a value of 0.0)
FREQUENCY_SCALE_FACTOR: '1.0'

C -- I/O processing should be on for applications run on the AG-100
I/O_PROCESSING: 'I/O PROCESSING ON'

C -- Alarms should be off unless the ALARM_WINDOW_PICTURE is present
ALARM_PROCESSING: 'ALARM PROCESSING OFF'

C -- The following is the first alarm picture to be loaded
ALARM_WINDOW_PICTURE: 'AQUA_SYS_ALARM.PIC'

C -- The following file defines the hardware I/O configuration
HARDWARE_CONNECTION_EDITOR_FILE: 'AQUA_SYS.IOC'

C -- The following line(s) define additional process picture files
C -- (Pictures may be chained using PROCESS and RETURN icons)
PROCESS_PICTURES: 'EXP1.PIC'
```

AQUARIUM CONTROL SYSTEM - Experiment 1

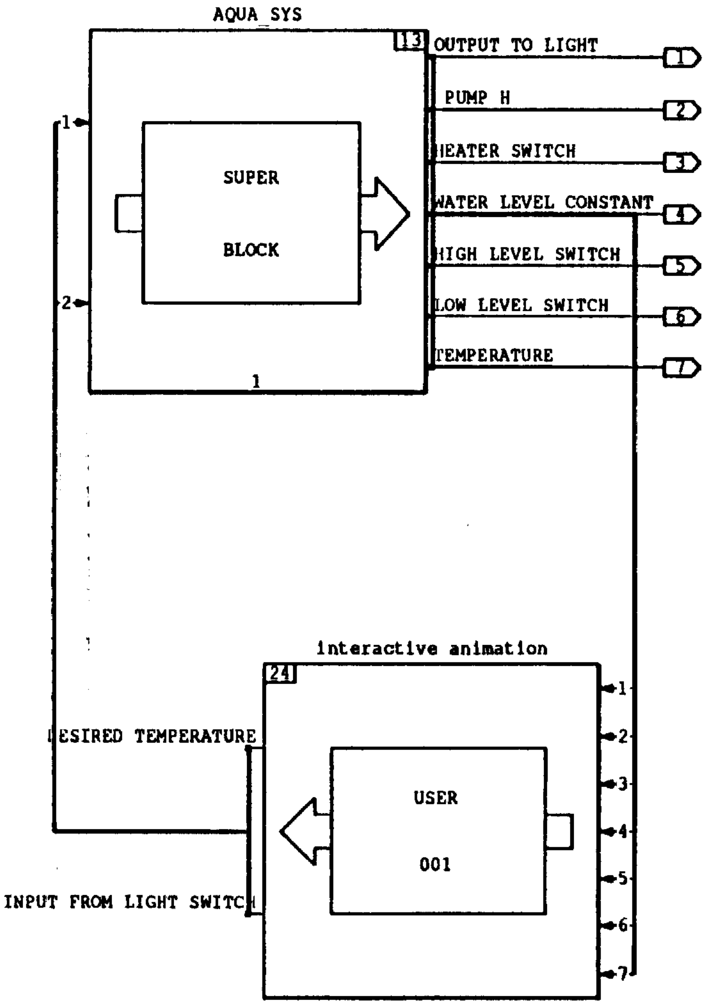


Expl.PIC
Page 1 of 1

Simulation File
Aqua_Sys.PS
Page 1 of 4

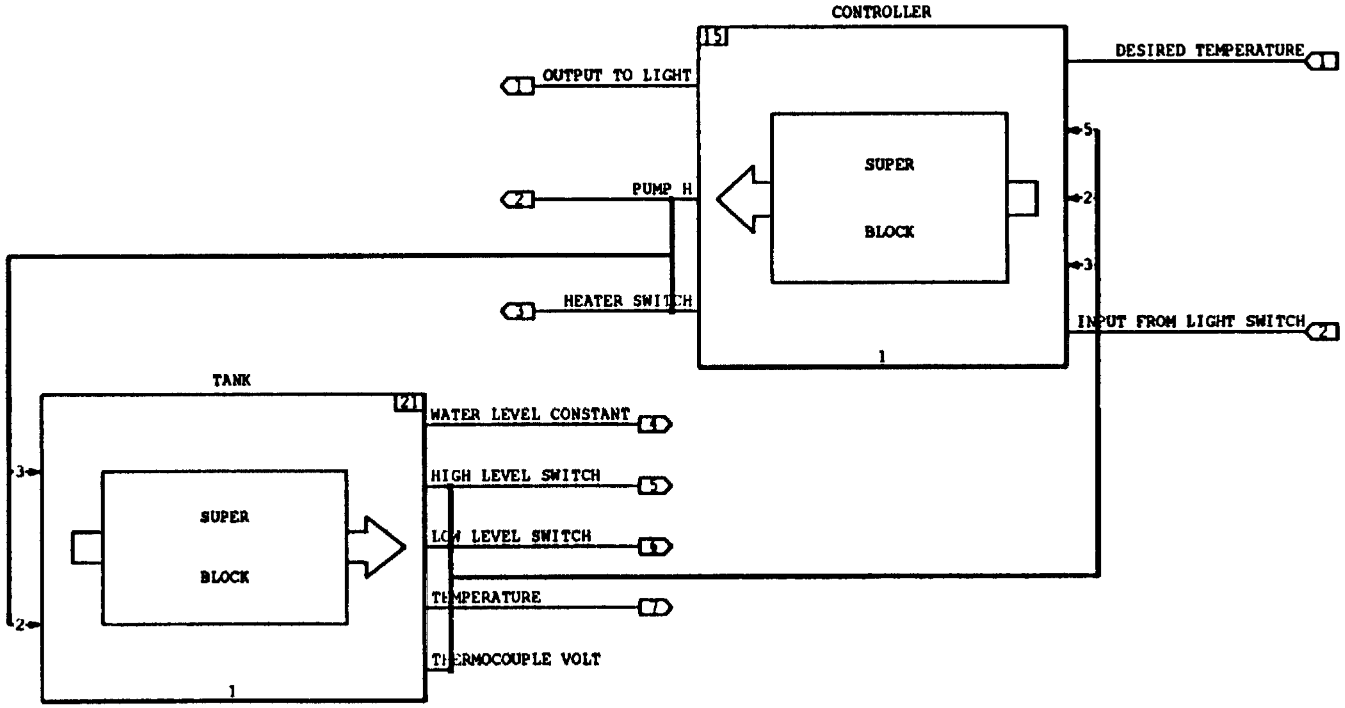
12-FEB-91

Discrete Super-Block	Sampling Interval	First Sample	Ext.Inputs	Ext.Outputs	Enable
system	1.	0.	0	7	Parent



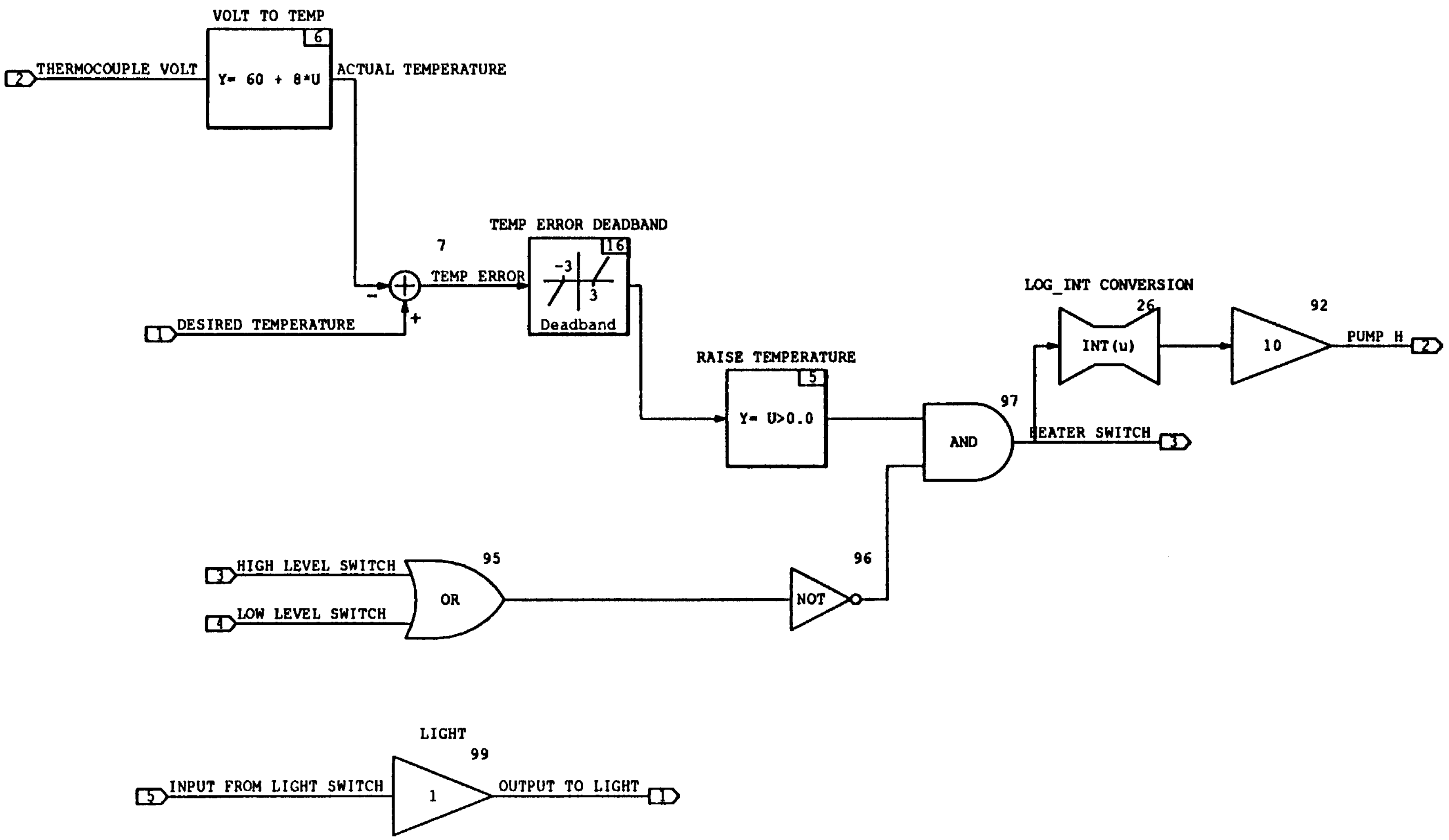
12-FEB-91

Discrete Super-Block	Sampling Interval	First Sample	Ext.Inputs	Ext.Outputs	Enable
AQUA SYS	1.	0.	2	7	Parent



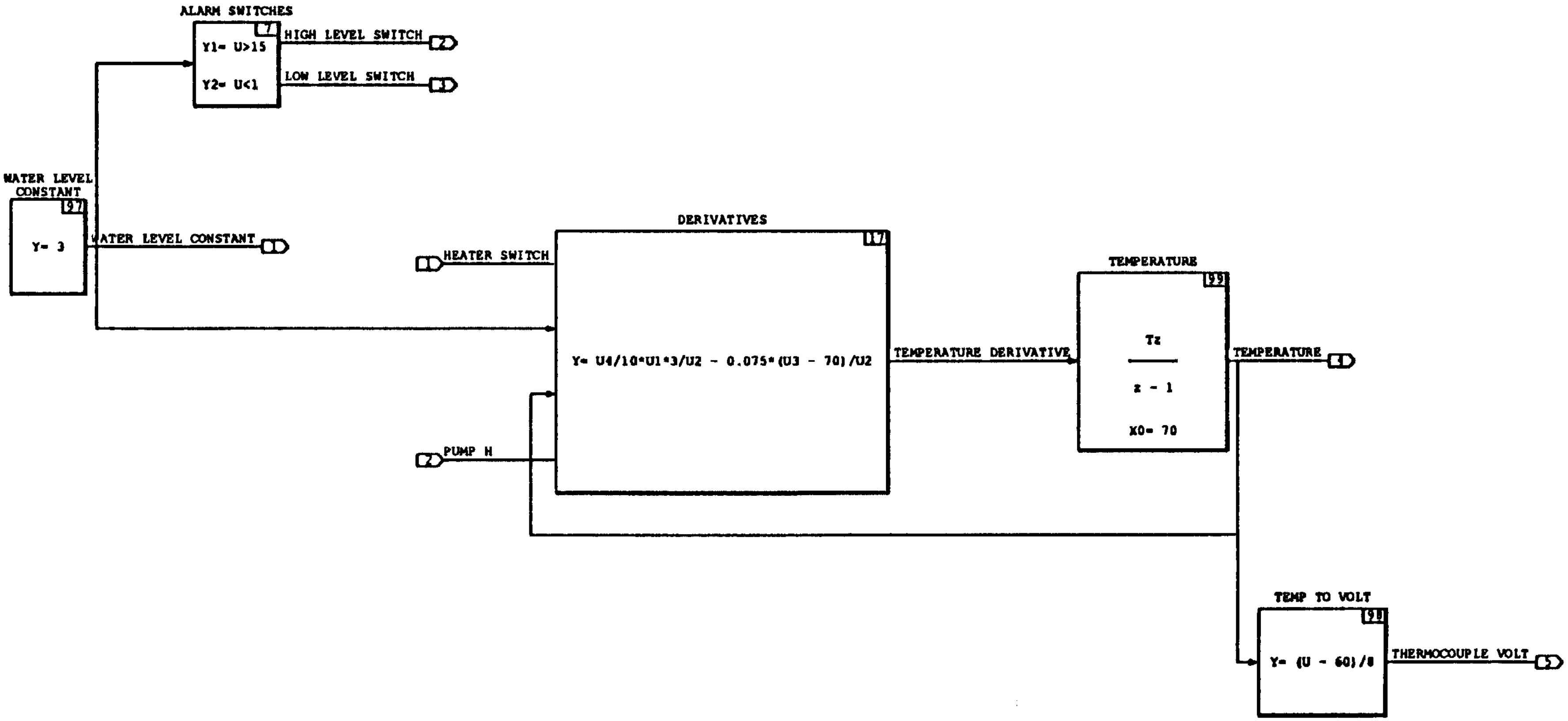
12-FEB-91

Discrete Super-Block	Sampling Interval	First Sample	Ext.Inputs	Ext.Outputs	Enable
CONTROLLER	1.	0.	5	3	Parent

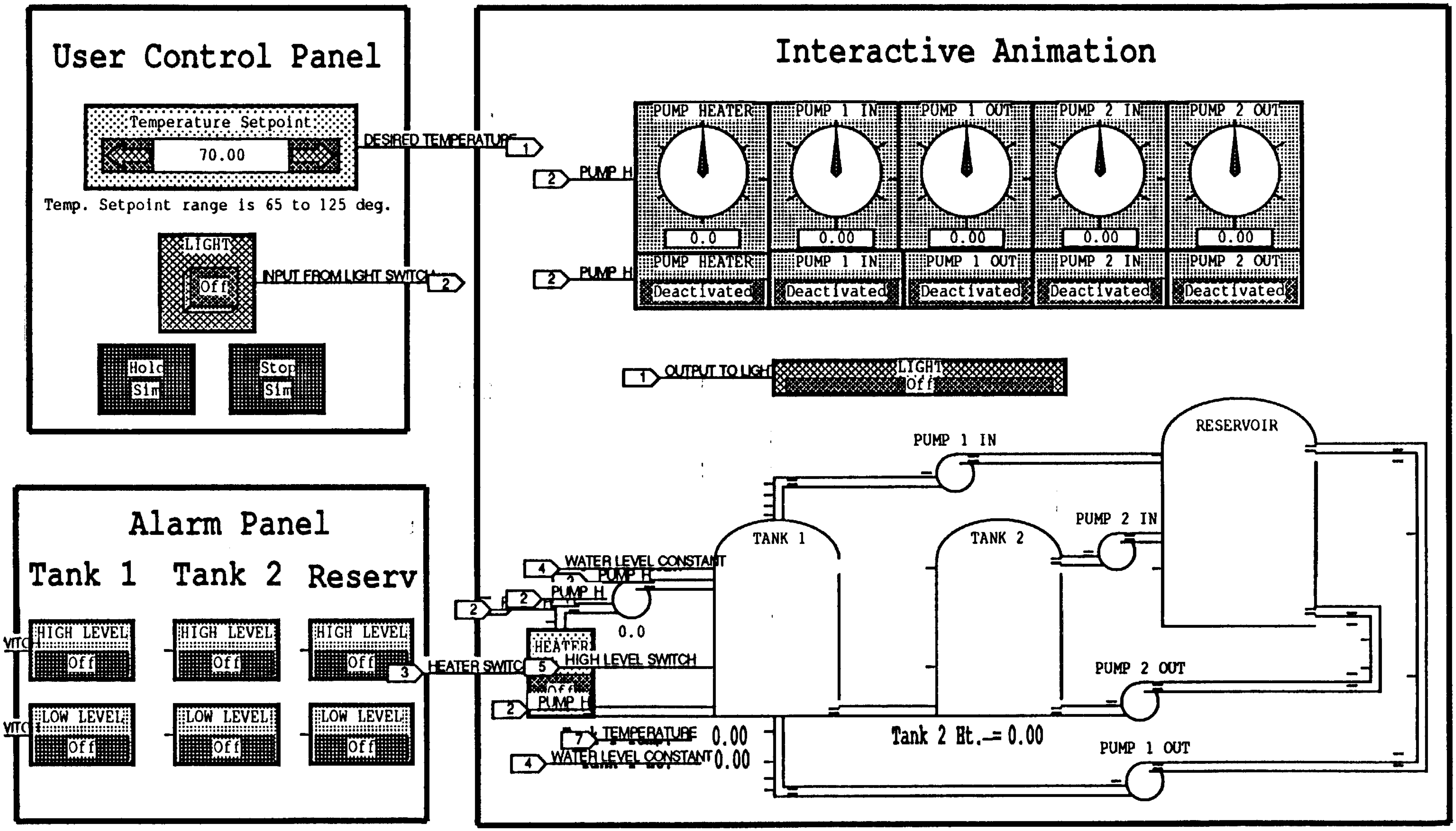


12-FEB-91

Discrete Super-Block	Sampling Interval	First Sample	Ext.Inputs	Ext.Outputs	Enable
TANK	1.	0.	2	5	Parent



AQUARIUM CONTROL SYSTEM - Experiment 1



```
RTMPG INPUT-OUTPUT CONFIGURATION FILE, VERSION 2.00
==============================================================================
   2 = NUMBER OF SYSTEM BUILD (SB) INPUT CHANNELS
SB
CHAN   HARDWARE CONNECTIONS        IA      DATA ACQ.    SAMPLING              CH
     I/O TYPE          ADDR CHAN  TO FROM SET   PERIOD  I/O PERIOD    MAX     MI
---- ------------------------- -------- ------------ ---------- --------------
   1 MONITOR              0    0   0    1 OFF 0.000E+00  1.00      0.000E+00 0.000
   2 MONITOR              0    0   0    2 OFF 0.000E+00  1.00      0.000E+00 0.000
******************************************************************************
   7 = NUMBER OF SYSTEM BUILD (SB) OUTPUT CHANNELS
SB
CHAN   HARDWARE CONNECTIONS        IA      DATA ACQ.    SAMPLING              CH
     I/O TYPE          ADDR CHAN  TO FROM SET   PERIOD  I/O PERIOD    MAX     MI
---- ------------------------- -------- ------------ ---------- --------------
   1 MONITOR              0    0   1    0 OFF 0.000E+00  1.00      0.000E+00 0.000
   2 MONITOR              0    0   2    0 OFF 0.000E+00  1.00      0.000E+00 0.000
   3 MONITOR              0    0   3    0 OFF 0.000E+00  1.00      0.000E+00 0.000
   4 MONITOR              0    0   4    0 OFF 0.000E+00  1.00      0.000E+00 0.000
   5 MONITOR              0    0   5    0 OFF 0.000E+00  1.00      0.000E+00 0.000
   6 MONITOR              0    0   6    0 OFF 0.000E+00  1.00      0.000E+00 0.000
   7 MONITOR              0    0   7    0 OFF 0.000E+00  1.00      0.000E+00 0.000
```

```
---- AutoCode/Ada Code Generator V2.204 -----
--
--   Model File     :  AQUA_SYS.RTF
--   Model Date     :  13-FEB-91 16:47
--
--   Generated File:  AQUA_SYS.ADA
--
--   Number of External Inputs : 2
--   Number of External Outputs: 7
--
--   Scheduler Frequency:     1.00000
--
--   SUBSYSTEM    FREQUENCY    TIME_SKEW    OUTPUT_TIME    TASK_TYPE
--   ---------    ---------    ---------    -----------    ---------
--       1          1.00000      0.00000         *         PERIODIC


with SA_TYPES;                     use SA_TYPES;

package SYSTEM_DATA is

    type TASK_STATE_TYPE is (IDLE, RUNNING, BLOCKED);

    EPSILON         : constant RT_FLOAT   :=  9.537E-07;
    EPS             : constant RT_FLOAT   :=  4.0*EPSILON;
    LOWEST_PRIORITY : constant            :=  1;

    OK              : constant INTEGER    :=  0;
    STOP_BLOCK      : constant INTEGER    :=  1;
    MATH_ERROR      : constant INTEGER    :=  2;
    STOPPED         : constant INTEGER    :=  3;
    UCB_ERROR       : constant INTEGER    :=  4;
    TIME_OVERFLOW   : constant INTEGER    := -1;
    UNKNOWN_ERROR   : constant INTEGER    :=  100;

    SCHEDULER_FREQ  : constant RT_FLOAT   :=  1.0;
    BUSIZE          : constant RT_INTEGER :=  17;
    NTASKS          : constant RT_INTEGER :=  1;
    NUMIN           : constant RT_INTEGER :=  2;
    NUMOUT          : constant RT_INTEGER :=  7;
    UE_PTR          : constant RT_INTEGER :=  35;
    YE_PTR          : constant RT_INTEGER :=  38;

    BUS_OFFSET      : INTG_ARRAY(0..NTASKS) := (0..NTASKS => 0);
    ERROR_FLAG      : INTG_ARRAY(1..NTASKS) := (1..NTASKS => 0);
    EXIT_FLAG       : array(1..NTASKS) of BOOLEAN := (1..NTASKS => FALSE);
    TASK_STATE      : array(1..NTASKS) of TASK_STATE_TYPE;

    BUS             : REAL_ARRAY(1..44) :=
      (1..36 => -EPSILON, 37 => 0.0, 38..44 => -EPSILON);

    FREQ            : constant REAL_ARRAY(1..NTASKS) :=
      (1..NTASKS => 1.0);

end SYSTEM_DATA;

package body SYSTEM_DATA is
    LIMIT : RT_INTEGER;
begin
    -- Initialize States --
    BUS(1)  := 0.0;
    BUS(1)  := 70.0;

end SYSTEM_DATA;


------------------
-- SUBSYSTEM01 --
------------------
with SA_TYPES;                     use SA_TYPES;
with SA_MATH_LIB;                  use SA_MATH_LIB;
with SA_FLOAT_MATH_LIB;            use SA_FLOAT_MATH_LIB;
with SYSTEM_DATA;                  use SYSTEM_DATA;

package SUBSYSTEM01 is

    SUBSYSTEM01_PRIORITY   : constant := 29;

    task type SUBSYSTEM01_TYPE is
        pragma PRIORITY (SUBSYSTEM01_PRIORITY);
        entry Wakeup;
    end SUBSYSTEM01_TYPE;

    type SUBSYSTEM01_POINTER is access SUBSYSTEM01_TYPE;
    SUBSYSTEM_1  : SUBSYSTEM01_POINTER;

end SUBSYSTEM01;

package body SUBSYSTEM01 is

    SUBSYS_ID      : constant := 1;

    procedure Section0_01 is separate;
    procedure Section1_01 is separate;

    task body SUBSYSTEM01_TYPE is
    begin
       OUTER_LOOP:
       loop
          begin
          loop
             accept Wakeup;

             if EXIT_FLAG(SUBSYS_ID)  then
                exit OUTER_LOOP;
             end if;

             if BUS_OFFSET(SUBSYS_ID) = 0   then
                Section0_01;
             else
                Section1_01;
             end if;

             TASK_STATE(SUBSYS_ID) := IDLE;
          end loop;

          exception
             when SA_TYPES.EXIT_CONDITION  =>
                  ERROR_FLAG(SUBSYS_ID) := STOP_BLOCK;
             when NUMERIC_ERROR  =>
                  ERROR_FLAG(SUBSYS_ID) := MATH_ERROR;
             when OTHERS  =>
                  ERROR_FLAG(SUBSYS_ID) := UNKNOWN_ERROR;
          end;
       end loop OUTER_LOOP;
    end SUBSYSTEM01_TYPE;

end SUBSYSTEM01;


---------------
-- SCHEDULER --
---------------
with SA_TYPES;                     use SA_TYPES;
with SYSTEM_DATA;                  use SYSTEM_DATA;

package SCHEDULER_DATA is

    -- Scheduler Constants and Tables --

    type SUBSYSTEM_TYPE is (PERIODIC, ENABLED_PERIODIC, TRIGGERED_ANT,
                            TRIGGERED_ATR, TRIGGERED_SAF);

    type SUBSYSTEM_TYPE_ARRAY is array(1..NTASKS) of SUBSYSTEM_TYPE;
    type INITIAL_STATE_ARRAY  is array(1..NTASKS) of TASK_STATE_TYPE;

    SCHEDULER_ID           : constant INTEGER     := 0;
    OFFYE                  : constant RT_INTEGER  := 37;
    NUMID                  : constant RT_INTEGER  := 7;
    NUMENABLED             : constant RT_INTEGER  := 0;
    NUMTRIGGERED           : constant RT_INTEGER  := 0;
    NUMDSWRITERS           : constant RT_INTEGER  := 0;
    DS_REGISTERS_EXT       : constant RT_INTEGER  := 0;

    TASK_TYPE              : constant SUBSYSTEM_TYPE_ARRAY  :=
      (1..NTASKS => PERIODIC);
    INITIAL_TASK_STATE     : constant INITIAL_STATE_ARRAY   :=
      (1..NTASKS => IDLE);
    START_COUNT            : constant INTG_ARRAY(1..NTASKS) :=
      (1..NTASKS => 0);
    SCHEDULING_COUNT       : constant INTG_ARRAY(1..NTASKS) :=
      (1..NTASKS => 0);
    OUTPUT_COUNT           : constant INTG_ARRAY(1..NTASKS) :=
      (1..NTASKS => 0);
    SH_OFFSET              : constant INTG_ARRAY(1..NTASKS) :=
      (1..NTASKS => 44);
    SE_OFFSET              : constant INTG_ARRAY(1..NTASKS) :=
      (1..NTASKS => 7);
    NUMEXT                 : constant INTG_ARRAY(1..NTASKS) :=
      (1..NTASKS => 0);
    ID_EVENT               : constant INTG_ARRAY(1..NTASKS) :=
      (1..NTASKS => 0);
    LOC_EVENT              : constant INTG_ARRAY(1..NTASKS) :=
      (1..NTASKS => 0);
    DS_REGISTERS           : constant INTG_ARRAY(1..NTASKS) :=
      (1..NTASKS => 0);
    DS_OFFSET              : constant INTG_ARRAY(1..NTASKS) :=
      (1..NTASKS => 0);
    ID                     : constant INTG_ARRAY(1..NUMID)  :=
      (1, 1, 1, 1, 1, 1, 1);
    LOC                         : constant INTG_ARRAY(1..NUMID)   :=
      (17, 15, 13, 3, 4, 5, 2);

end SCHEDULER_DATA;


with SUBSYSTEM01;                           use SUBSYSTEM01;
with SA_UTILITIES;                          use SA_UTILITIES;
with SA_TYPES;                              use SA_TYPES;
with SCHEDULER_DATA;                        use SCHEDULER_DATA;
with SYSTEM_DATA;                           use SYSTEM_DATA;
with SA_IO_DEFINITIONS;

package body SA_SCHEDULER is

    type TCB_TYPE is
       record
         TASK_TYPE              : SUBSYSTEM_TYPE;
         ENABLED                : BOOLEAN;
         START                  : RT_INTEGER;
         START_COUNT            : RT_INTEGER;
         SCHEDULING_COUNT       : RT_INTEGER;
         ID_EVENT               : RT_INTEGER;
         LOC_EVENT              : RT_INTEGER;
         OUTPUT                 : RT_INTEGER;
         OUTPUT_COUNT           : RT_INTEGER;
         SH_OFFSET              : RT_INTEGER;
         SE_OFFSET              : RT_INTEGER;
         NUMEXT                 : RT_INTEGER;
         DS_OFFSET              : RT_INTEGER;
         DS_REGISTERS           : RT_INTEGER;
       end record;

    TSK                         : RT_INTEGER;
    OFFSE                       : RT_INTEGER;
    OFFSH                       : RT_INTEGER;
    OFFDS                       : RT_INTEGER;
    READY_COUNT                 : RT_INTEGER;
    DISPATCH_COUNT              : RT_INTEGER;
    READY_QUEUE                 : INTG_ARRAY(1..NTASKS);
    DISPATCH                    : BOOLEAN_ARRAY(1..NTASKS);
    TCB                         : array(1..NTASKS) of TCB_TYPE;


    procedure Queue_Task(NTSK : in RT_INTEGER) is
    begin
       READY_COUNT                   := READY_COUNT + 1;
       READY_QUEUE(READY_COUNT)      := NTSK;
       DISPATCH(NTSK)                := true;
       TASK_STATE(NTSK)              := RUNNING;
    end Queue_Task;
    pragma inline (Queue_Task);


    procedure Update_Outputs(NTSK : in RT_INTEGER) is
    begin
       BUS_OFFSET(NTSK)  := BUSIZE - BUS_OFFSET(NTSK);
    end Update_Outputs;
    pragma inline (Update_Outputs);
```

```
procedure Signal_An_Error(NTSK : in RT_INTEGER) is
begin
   if ERROR_FLAG(NTSK) = OK  then
      ERROR( NTSK, TIME_OVERFLOW );
   else
      ERROR( NTSK, ERROR_FLAG(NTSK) );
   end if;
end Signal_An_Error;


procedure Init_Scheduler is
begin
   for NTSK in 1..NTASKS  loop
      TCB(NTSK).TASK_TYPE        := TASK_TYPE(NTSK);
      TCB(NTSK).ENABLED          := false;
      TCB(NTSK).START            := START_COUNT(NTSK);
      TCB(NTSK).START_COUNT      := START_COUNT(NTSK);
      TCB(NTSK).SCHEDULING_COUNT := SCHEDULING_COUNT(NTSK);
      TCB(NTSK).ID_EVENT         := ID_EVENT(NTSK);
      TCB(NTSK).LOC_EVENT        := LOC_EVENT(NTSK);
      TCB(NTSK).OUTPUT           := OUTPUT_COUNT(NTSK);
      TCB(NTSK).OUTPUT_COUNT     := OUTPUT_COUNT(NTSK);
      TCB(NTSK).SH_OFFSET        := SH_OFFSET(NTSK);
      TCB(NTSK).SE_OFFSET        := SE_OFFSET(NTSK);
      TCB(NTSK).NUMEXT           := NUMEXT(NTSK);
      TCB(NTSK).DS_OFFSET        := DS_OFFSET(NTSK);
      TCB(NTSK).DS_REGISTERS     := DS_REGISTERS(NTSK);
      TASK_STATE(NTSK)           := INITIAL_TASK_STATE(NTSK);
      DISPATCH(NTSK)             := false;
      if TASK_TYPE(NTSK)  = TRIGGERED_ATR or
         TASK_TYPE(NTSK)  = TRIGGERED_SAF then
         BUS_OFFSET(NTSK)  := 0;
      else
         BUS_OFFSET(NTSK)  := BUSIZE;
      end if;
   end loop;
   DISPATCH_COUNT   := 0;
   READY_COUNT      := 0;
   READY_QUEUE(1)   := 0;
   SCHEDULER_STATUS := OK;
   INTERRUPT_COUNT  := 0;
end Init_Scheduler;


task body SCHEDULER_TYPE is
begin
  SCHEDULER_LOOP:
  loop
    INNER_LOOP:
    loop

      -- Scheduler Wakeup --

      accept Wakeup;

      INTERRUPT_COUNT := INTERRUPT_COUNT + 1;
      SA_IO_DEFINITIONS.STATUS_386.SCHEDULER_READY_INDEX := 0; -- False
```

```
if SCHEDULER_STATUS /= OK  then
   ERROR(SCHEDULER_ID, SCHEDULER_STATUS);
   exit INNER_LOOP;
end if;

-- System Input --

External_Input( BUS, UE_PTR, NUMIN, SCHEDULER_STATUS );

-- Task Scheduling --

for NTSK in reverse 1..NTASKS  loop

   case TASK_STATE(NTSK) is
      when IDLE =>

         case TCB(NTSK).TASK_TYPE is
            when PERIODIC =>
               if TCB(NTSK).START = 0 then
                  Queue_Task(NTSK);
                  Update_Outputs(NTSK);
                  TCB(NTSK).START   := TCB(NTSK).SCHEDULING_COUNT;
               else
                  TCB(NTSK).START   := TCB(NTSK).START - 1;
               end if;

            when ENABLED_PERIODIC =>
               if not TCB(NTSK).ENABLED then
                  TASK_STATE(NTSK)  := BLOCKED;
               elsif TCB(NTSK).START = 0 then
                  Queue_Task(NTSK);
                  Update_Outputs(NTSK);
                  TCB(NTSK).START   := TCB(NTSK).SCHEDULING_COUNT;
               else
                  TCB(NTSK).START   := TCB(NTSK).START - 1;
               end if;

            when TRIGGERED_ANT =>
               if TCB(NTSK).START = 0 then
                  Queue_Task(NTSK);
                  Update_Outputs(NTSK);
                  TCB(NTSK).START   := 1;
               end if;

            when TRIGGERED_ATR =>
               if TCB(NTSK).OUTPUT = 0 then
                  Update_Outputs(NTSK);
                  TASK_STATE(NTSK)  := BLOCKED;
               else
                  TCB(NTSK).OUTPUT := TCB(NTSK).OUTPUT - 1;
               end if;
               if TCB(NTSK).START = 0 then
                  Queue_Task(NTSK);
                  TCB(NTSK).OUTPUT := TCB(NTSK).OUTPUT_COUNT;
                  TCB(NTSK).START   := 1;
               end if;

            when TRIGGERED_SAF =>
               if TCB(NTSK).OUTPUT = 0 then
                  Update_Outputs(NTSK);
            TASK_STATE(NTSK) := BLOCKED;
         end if;
         if TCB(NTSK).START = 0 then
            Queue_Task(NTSK);
            TCB(NTSK).OUTPUT := 0;
            TCB(NTSK).START  := 1;
         end if;

      when OTHERS =>
         null;
   end case;

when RUNNING =>

   case TCB(NTSK).TASK_TYPE is
      when PERIODIC =>
         if TCB(NTSK).START > 0 then
            TCB(NTSK).START  := TCB(NTSK).START - 1;
         else
            Signal_An_Error(NTSK);
            exit INNER_LOOP;
         end if;

      when ENABLED_PERIODIC =>
         if TCB(NTSK).START > 0 then
            TCB(NTSK).START  := TCB(NTSK).START - 1;
         else
            Signal_An_Error(NTSK);
            exit INNER_LOOP;
         end if;

      when TRIGGERED_ANT =>
         if TCB(NTSK).START = 0 then
            Signal_An_Error(NTSK);
            exit INNER_LOOP;
         end if;

      when TRIGGERED_ATR =>
         if TCB(NTSK).OUTPUT > 0 then
            TCB(NTSK).OUTPUT := TCB(NTSK).OUTPUT - 1;
         else
            Signal_An_Error(NTSK);
            exit INNER_LOOP;
         end if;

      when TRIGGERED_SAF =>
         if ERROR_FLAG(NTSK) /= 0 then
            Signal_An_Error(NTSK);
            exit INNER_LOOP;
         end if;

      when OTHERS =>
         null;
   end case;

when BLOCKED =>

   case TCB(NTSK).TASK_TYPE is
      when ENABLED_PERIODIC =>
         if TCB(NTSK).ENABLED then
```

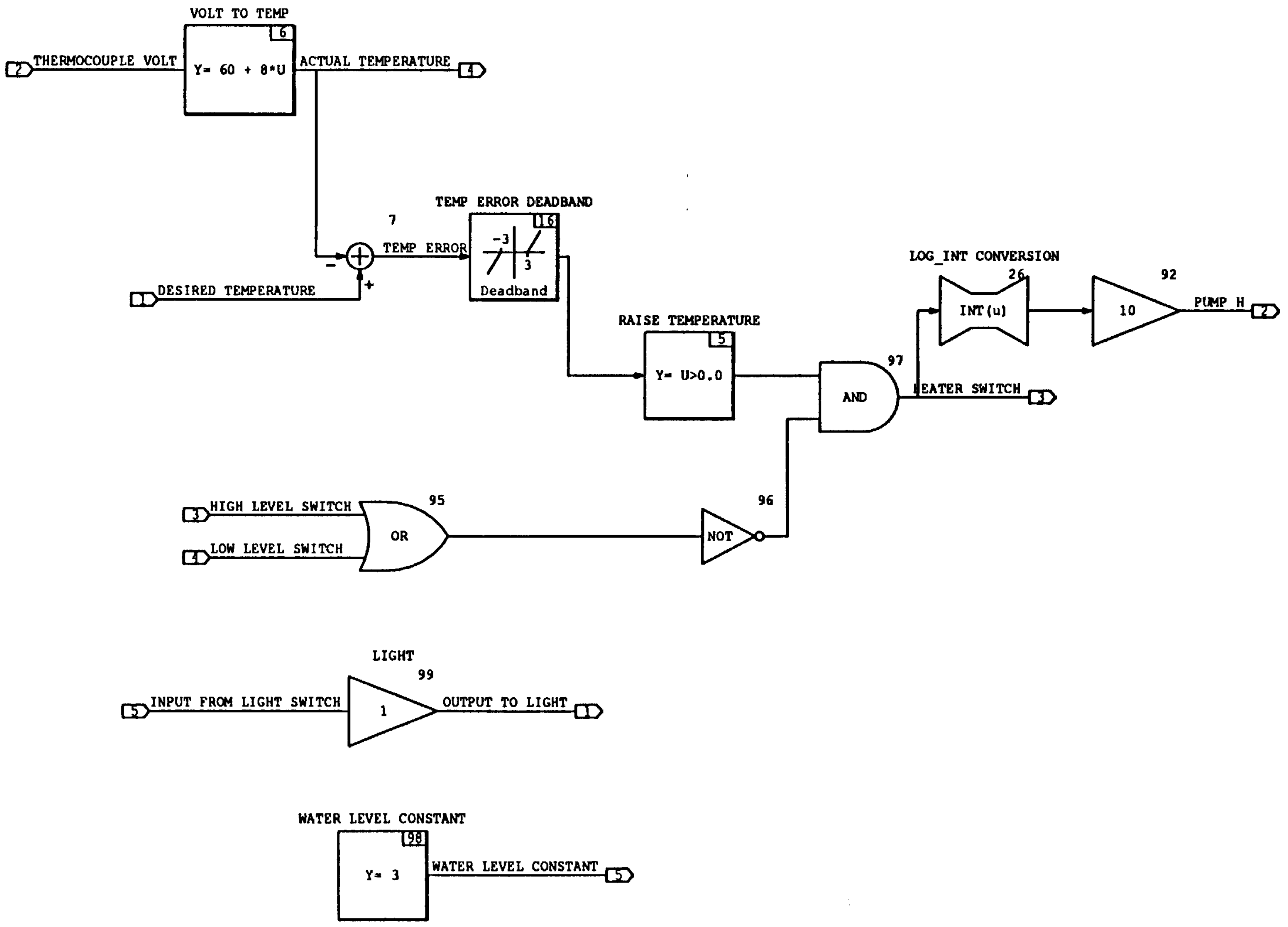
```
                  Queue_Task(NTSK);
                  Update_Outputs(NTSK);
                  TCB(NTSK).START  := TCB(NTSK).SCHEDULING_COUNT;
               end if;

            when TRIGGERED_ATR =>
               if TCB(NTSK).START = 0 then
                  Queue_Task(NTSK);
                  TCB(NTSK).OUTPUT := TCB(NTSK).OUTPUT_COUNT;
                  TCB(NTSK).START  := 1;
               end if;

            when TRIGGERED_SAF =>
               if TCB(NTSK).START = 0 then
                  Queue_Task(NTSK);
                  TCB(NTSK).OUTPUT := 0;
                  TCB(NTSK).START  := 1;
               end if;

            when OTHERS =>
               null;
         end case;

   end case;
end loop;

-- System Output --

for J in 1..NUMOUT    loop
   BUS(J+OFFYE) := BUS( LOC(J)+BUS_OFFSET(ID(J)) );
end loop;

External_Output ( BUS, YE_PTR, NUMOUT, SCHEDULER_STATUS );

-- Task Input Sample and Hold --

for I in reverse 1..READY_COUNT  loop
   TSK   := READY_QUEUE(I);
   OFFSE := TCB(TSK).SE_OFFSET;
   OFFSH := TCB(TSK).SH_OFFSET - OFFSE;
   for J in OFFSE+1..OFFSE+TCB(TSK).NUMEXT   loop
      BUS(J+OFFSH) := BUS( LOC(J)+BUS_OFFSET(ID(J)) );
   end loop;
end loop;

-- Clear Ready Queue --

if READY_QUEUE(1) > DISPATCH_COUNT then
   DISPATCH_COUNT := READY_QUEUE(1);
end if;
READY_COUNT    := 0;
READY_QUEUE(1) := 0;

-- Task Dispatching --

for NTSK in 1..DISPATCH_COUNT  loop
   if DISPATCH(NTSK) then
      DISPATCH(NTSK) := false;
      case NTSK is
         when 1       =>  SUBSYSTEM_1.Wakeup;
```

```
                   when OTHERS  =>  null;
                end case;
             end if;
          end loop;
          DISPATCH_COUNT := 0;

          INTERRUPT_COUNT := INTERRUPT_COUNT - 1;
          SA_IO_DEFINITIONS.STATUS_386.SCHEDULER_READY_INDEX := 1;  -- True

        end loop INNER_LOOP;

        exit;

      end loop SCHEDULER_LOOP;

      -- Shutdown --

      accept Quit;

    end SCHEDULER_TYPE;
end SA_SCHEDULER;


----------
-- MAIN --
----------
with SUBSYSTEM01;                  use SUBSYSTEM01;
with SA_MPC_INTERRUPT;             use SA_MPC_INTERRUPT;
with SA_IO_DEFINITIONS;            use SA_IO_DEFINITIONS;
with SA_SCHEDULER;                 use SA_SCHEDULER;
with SA_UTILITIES;                 use SA_UTILITIES;
with SYSTEM_DATA;                  use SYSTEM_DATA;

procedure AQUA_SYS is

    ------------------
    -- COMMAND_TASK --
    ------------------
    task type COMMAND_TASK_TYPE is
        pragma PRIORITY (LOWEST_PRIORITY);
    end COMMAND_TASK_TYPE;

    type COMMAND_TASK_POINTER is access COMMAND_TASK_TYPE;
    COMMAND_TASK  : COMMAND_TASK_POINTER;

    task body COMMAND_TASK_TYPE is
    begin
        while STATUS_386.TASK_COMMAND /= CONVERT_INPUT_DATA    loop
          for WAIT_CNT in 1..10    loop
             null;
          end loop;
        end loop;
        SCHEDULER_STATUS := STOPPED;
        SCHEDULER.Wakeup;
    end COMMAND_TASK_TYPE;

    procedure Shutdown_The_Tasks is
    begin
        for NTSK in 1..NTASKS  loop
          EXIT_FLAG(NTSK) := true;
        end loop;
        SUBSYSTEM_1.Wakeup;
    end Shutdown_The_Tasks;

begin

    -- Create Subsystem Tasks --

    SUBSYSTEM_1  := new SUBSYSTEM01_TYPE;

    -- Start Scheduler --

    Init_Scheduler;
    SCHEDULER := new SCHEDULER_TYPE;

    -- User Initialization --

    Implementation_Specific_Initialize (BUS, UE_PTR, NUMIN, YE_PTR, NUMOUT,
                                        SCHEDULER_STATUS, SCHEDULER_FREQ);

    if  SCHEDULER_STATUS = OK  then
        COMMAND_TASK := new COMMAND_TASK_TYPE;
    else
        SCHEDULER.Wakeup;
    end if;

    -- User Termination --

    SCHEDULER.Quit;

    if  STATUS_386.TASK_COMMAND = CONVERT_INPUT_DATA   then
        Implementation_Specific_Terminate;
    else
        STATUS_386.ERRORTYPE := 1; -- Reboot controller
    end if;

    Shutdown_The_Tasks;

end AQUA_SYS;



separate (SUBSYSTEM01)
    procedure Section0_01 is
    begin

    ------------------------------------ Nth Order Integrator
    -- {TANK.TEMPERATURE.99}
        BUS(19)  := 1.0 * BUS(33) + BUS(1);
        BUS(19)  := 1.0 * BUS(19);
    ------------------------------------ General Nested Expression
    -- {TANK.WATER LEVEL CONSTANT.97}
        BUS(20)  := 3.0;
    ------------------------------------ General Logical Expression
    -- {TANK.ALARM SWITCHES.7}
        if BUS(20)>15.0 then
            BUS(21)  := 1.0;
        else
            BUS(21)  := 0.0;
        end if;
    if BUS(20)<1.0 then
        BUS(22) := 1.0;
    else
        BUS(22) := 0.0;
    end if;
------------------------------------ OR   Logical Block
-- {CONTROLLER..95}
    if  BUS(21) > 0.0 or  BUS(22) > 0.0  then
        BUS(23) := 1.0;
    else
        BUS(23) := 0.0;
    end if;
------------------------------------ NOT  Logical Block
-- {CONTROLLER..96}
    if  BUS(23) > 0.0  then
        BUS(24) := 0.0;
    else
        BUS(24) := 1.0;
    end if;
------------------------------------ General Nested Expression
-- {TANK.TEMP TO VOLT.98}
    BUS(25) := ( BUS(19) - 60.0 )/8.0;
------------------------------------ General Nested Expression
-- {CONTROLLER.VOLT TO TEMP.6}
    BUS(26) := 60.0 + 8.0*BUS(25);
------------------------------------ Summing Junction
-- {CONTROLLER..7}
    BUS(27) := -BUS(26) + BUS(35);
------------------------------------ Dead Band
-- {CONTROLLER.TEMP ERROR DEADBAND.16}
    BUS(28) := BUS(27) - 3.0;
    if BUS(28) < 0.0  then
       BUS(28) := BUS(27) + 3.0;
       if BUS(28) > 0.0  then
          BUS(28) := 0.0;
       end if;
    end if;
------------------------------------ General Logical Expression
-- {CONTROLLER.RAISE TEMPERATURE.5}
    if BUS(28)>0.0 then
        BUS(29) := 1.0;
    else
        BUS(29) := 0.0;
    end if;
------------------------------------ AND  Logical Block
-- {CONTROLLER..97}
    if  BUS(29) > 0.0 and BUS(24) > 0.0  then
        BUS(30) := 1.0;
    else
        BUS(30) := 0.0;
    end if;
------------------------------------ Data Conversion
-- {CONTROLLER.LOG_INT CONVERSION.26}
    BUS(31) := RT_FLOAT(RT_INTEGER(BUS(30)));
------------------------------------ Gain Block
-- {CONTROLLER..92}
    BUS(32) := 10.0 * BUS(31);
------------------------------------ General Nested Expression
-- {TANK.DERIVATIVES.17}
    BUS(33) := BUS(32)/10.0*BUS(30)*3.0/BUS(20) - 0.075*( BUS(19)
                    - 70.0 )/BUS(20);
    ------------------------------------  Gain Block
    -- {CONTROLLER.LIGHT.99}
        BUS(34) := BUS(36);
    -- *
    ------------------------------------  Nth Order Integrator
    -- {TANK.TEMPERATURE.99}
        BUS(18) := 1.0 * BUS(33) + BUS(1);

    end Section0_01;


separate (SUBSYSTEM01)
    procedure Section1_01 is
    begin

    ------------------------------------  Nth Order Integrator
    -- {TANK.TEMPERATURE.99}
        BUS(2) := 1.0 * BUS(16) + BUS(18);
        BUS(2) := 1.0 * BUS(2);
    ------------------------------------  General Nested Expression
    -- {TANK.WATER LEVEL CONSTANT.97}
        BUS(3) := 3.0;
    ------------------------------------  General Logical Expression
    -- {TANK.ALARM SWITCHES.7}
        if BUS(3)>15.0 then
            BUS(4) := 1.0;
        else
            BUS(4) := 0.0;
        end if;
        if BUS(3)<1.0 then
            BUS(5) := 1.0;
        else
            BUS(5) := 0.0;
        end if;
    ------------------------------------  OR   Logical Block
    -- {CONTROLLER..95}
        if  BUS(4) > 0.0 or  BUS(5) > 0.0  then
            BUS(6) := 1.0;
        else
            BUS(6) := 0.0;
        end if;
    ------------------------------------  NOT  Logical Block
    -- {CONTROLLER..96}
        if  BUS(6) > 0.0  then
            BUS(7) := 0.0;
        else
            BUS(7) := 1.0;
        end if;
    ------------------------------------  General Nested Expression
    -- {TANK.TEMP TO VOLT.98}
        BUS(8) := ( BUS(2) - 60.0 )/8.0;
    ------------------------------------  General Nested Expression
    -- {CONTROLLER.VOLT TO TEMP.6}
        BUS(9) := 60.0 + 8.0*BUS(8);
    ------------------------------------  Summing Junction
    -- {CONTROLLER..7}
        BUS(10) := -BUS(9) + BUS(35);
    ------------------------------------  Dead Band
```

```
-- {CONTROLLER.TEMP ERROR DEADBAND.16}
    BUS(11) := BUS(10) - 3.0;
    if BUS(11) < 0.0  then
       BUS(11) := BUS(10) + 3.0;
       if BUS(11) > 0.0  then
          BUS(11) := 0.0;
       end if;
    end if;
------------------------------------  General Logical Expression
-- {CONTROLLER.RAISE TEMPERATURE.5}
    if BUS(11)>0.0 then
        BUS(12) := 1.0;
    else
        BUS(12) := 0.0;
    end if;
------------------------------------  AND  Logical Block
-- {CONTROLLER..97}
    if  BUS(12) > 0.0 and BUS(7) > 0.0  then
        BUS(13) := 1.0;
    else
        BUS(13) := 0.0;
    end if;
------------------------------------  Data Conversion
-- {CONTROLLER.LOG_INT CONVERSION.26}
    BUS(14) := RT_FLOAT(RT_INTEGER(BUS(13)));
------------------------------------  Gain Block
-- {CONTROLLER..92}
    BUS(15) := 10.0 * BUS(14);
------------------------------------  General Nested Expression
-- {TANK.DERIVATIVES.17}
    BUS(16) := BUS(15)/10.0*BUS(13)*3.0/BUS(3) - 0.075*( BUS(2) -
                70.0 )/BUS(3);
------------------------------------  Gain Block
-- {CONTROLLER.LIGHT.99}
    BUS(17) := BUS(36);
-- *
------------------------------------  Nth Order Integrator
-- {TANK.TEMPERATURE.99}
    BUS(1) := 1.0 * BUS(16) + BUS(18);

end Section1_01;
```

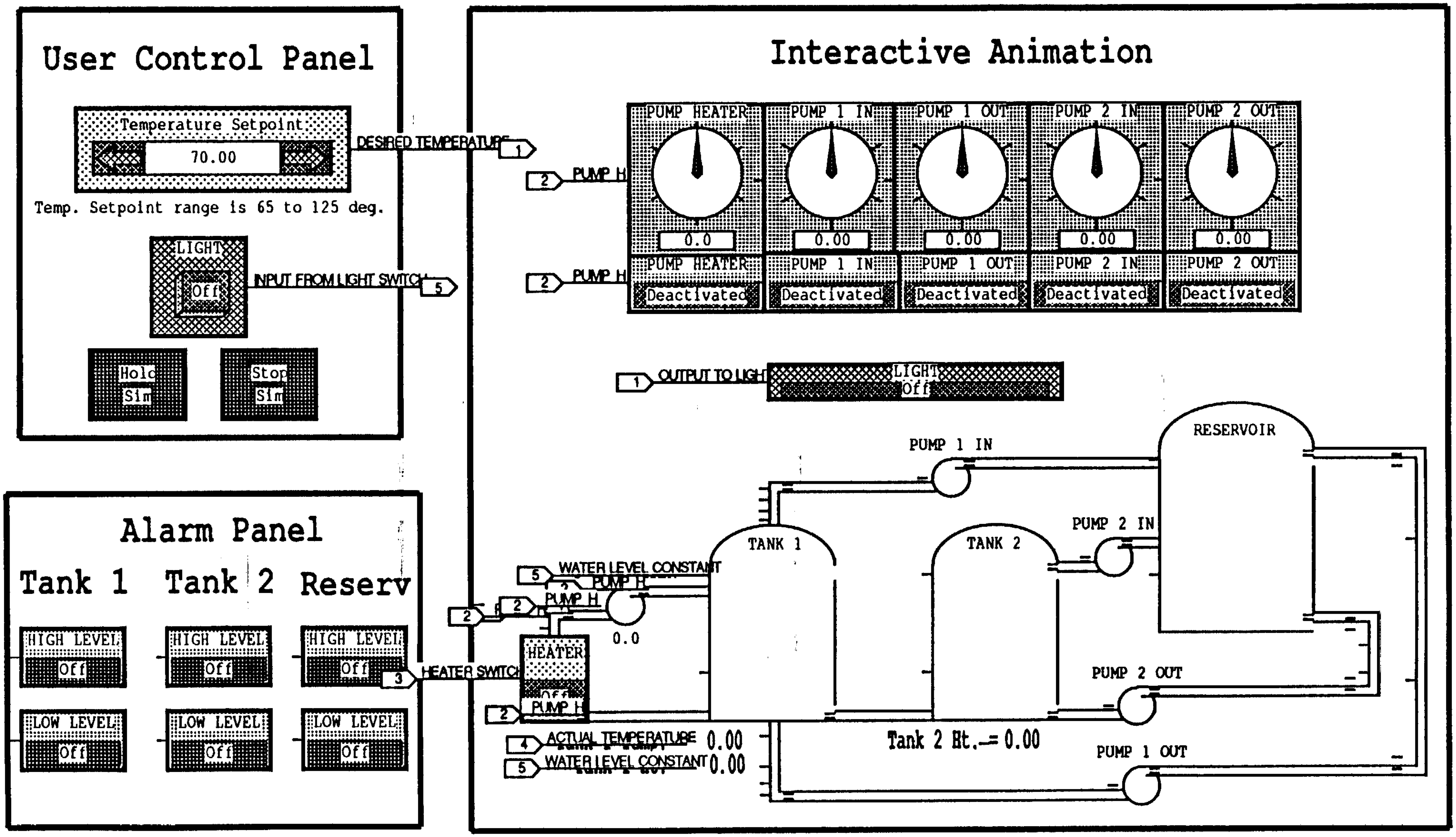
13-FEB-91

Discrete Super-Block	Sampling Interval	First Sample	Ext.Inputs	Ext.Outputs	Enable
CONTROLLER	1.	0.	5	5	Parent



HW IF File
Controller.PS
Page 1 of 1

AQUARIUM CONTROL SYSTEM - Experiment 1



User Control Panel

Temperature Setpoint: 70.00

Temp. Setpoint range is 65 to 125 deg.

Interactive Animation

Alarm Panel

HW IF File
Aqua_Sys.IOC
Page 1 of 1

```
RTMPG INPUT-OUTPUT CONFIGURATION FILE, VERSION 2.00
==============================================================================
   5 = NUMBER OF SYSTEM BUILD (SB) INPUT CHANNELS
SB
CHAN   HARDWARE CONNECTIONS      IA      DATA ACQ.    SAMPLING               CH
     I/O TYPE        ADDR CHAN  TO FROM SET   PERIOD  I/O PERIOD     MAX     MI
---- ------------------------ -------- ------------ ---------- --------------
   1 MONITOR            0    0   0     1 OFF 0.000E+00  1.00     0.000E+00 0.000
   2 OPTO_AD12          8    1   0     0 OFF 0.000E+00  1.00      10.0     -10.0
   3 OPTO_IDC5BQ        4    1   0     0 OFF 0.000E+00  1.00      1.00     0.000
   4 OPTO_IDC5BQ        4    2   0     0 OFF 0.000E+00  1.00      1.00     0.000
   5 MONITOR            0    0   0     2 OFF 0.000E+00  1.00     0.000E+00 0.000
******************************************************************************
   5 = NUMBER OF SYSTEM BUILD (SB) OUTPUT CHANNELS
SB
CHAN   HARDWARE CONNECTIONS      IA      DATA ACQ.    SAMPLING               CH
     I/O TYPE        ADDR CHAN  TO FROM SET   PERIOD  I/O PERIOD     MAX     MI
---- ------------------------ -------- ------------ ---------- --------------
   1 OPTO_ODC5AQ        4   17   1     0 OFF 0.000E+00  1.00      1.00     0.000
   2 OPTO_DA7           8    9   2     0 OFF 0.000E+00  1.00      10.0     -10.0
   3 OPTO_ODC5AQ        4   18   3     0 OFF 0.000E+00  1.00      1.00     0.000
   4 OPTO_DA7           8   10   4     0 OFF 0.000E+00  1.00      10.0     -10.0
   5 MONITOR            0    0   5     0 OFF 0.000E+00  1.00     0.000E+00 0.000
```

```
---- AutoCode/Ada Code Generator V2.204 -----
--
--   Model File    :  AQUA_SYS.RTF
--   Model Date    :  13-FEB-91 18:10
--
--   Generated File:  AQUA_SYS.ADA
--
--   Number of External Inputs : 5
--   Number of External Outputs: 5
--
--   Scheduler Frequency:     1.00000
--
--   SUBSYSTEM    FREQUENCY    TIME_SKEW    OUTPUT_TIME    TASK_TYPE
--   ---------    ---------    ---------    -----------    ---------
--       1          1.00000      0.00000         *         PERIODIC


with SA_TYPES;                    use SA_TYPES;

package SYSTEM_DATA is

    type TASK_STATE_TYPE is (IDLE, RUNNING, BLOCKED);

    EPSILON         : constant RT_FLOAT   :=  9.537E-07;
    EPS             : constant RT_FLOAT   :=  4.0*EPSILON;
    LOWEST_PRIORITY : constant            :=  1;

    OK              : constant INTEGER    :=  0;
    STOP_BLOCK      : constant INTEGER    :=  1;
    MATH_ERROR      : constant INTEGER    :=  2;
    STOPPED         : constant INTEGER    :=  3;
    UCB_ERROR       : constant INTEGER    :=  4;
    TIME_OVERFLOW   : constant INTEGER    := -1;
    UNKNOWN_ERROR   : constant INTEGER    :=  100;

    SCHEDULER_FREQ  : constant RT_FLOAT   :=  1.0;
    BUSIZE          : constant RT_INTEGER :=  11;
    NTASKS          : constant RT_INTEGER :=  1;
    NUMIN           : constant RT_INTEGER :=  5;
    NUMOUT          : constant RT_INTEGER :=  5;
    UE_PTR          : constant RT_INTEGER :=  23;
    YE_PTR          : constant RT_INTEGER :=  29;

    BUS_OFFSET      : INTG_ARRAY(0..NTASKS)  := (0..NTASKS => 0);
    ERROR_FLAG      : INTG_ARRAY(1..NTASKS)  := (1..NTASKS => 0);
    EXIT_FLAG       : array(1..NTASKS) of BOOLEAN := (1..NTASKS => FALSE);
    TASK_STATE      : array(1..NTASKS) of TASK_STATE_TYPE;

    BUS             : REAL_ARRAY(1..33) :=
      (1..27 => -EPSILON, 28 => 0.0, 29..33 => -EPSILON);

    FREQ            : constant REAL_ARRAY(1..NTASKS) :=
      (1..NTASKS => 1.0);

end SYSTEM_DATA;

package body SYSTEM_DATA is
    LIMIT : RT_INTEGER;
begin
    null;
```

```
end SYSTEM_DATA;


------------------
-- SUBSYSTEM01 --
------------------
with SA_TYPES;                     use SA_TYPES;
with SA_MATH_LIB;                  use SA_MATH_LIB;
with SA_FLOAT_MATH_LIB;            use SA_FLOAT_MATH_LIB;
with SYSTEM_DATA;                  use SYSTEM_DATA;

package SUBSYSTEM01 is

    SUBSYSTEM01_PRIORITY   : constant := 29;

    task type SUBSYSTEM01_TYPE is
        pragma PRIORITY (SUBSYSTEM01_PRIORITY);
        entry Wakeup;
    end SUBSYSTEM01_TYPE;

    type SUBSYSTEM01_POINTER is access SUBSYSTEM01_TYPE;
    SUBSYSTEM_1  : SUBSYSTEM01_POINTER;

end SUBSYSTEM01;

package body SUBSYSTEM01 is

    SUBSYS_ID      : constant := 1;

    procedure Section0_01 is separate;
    procedure Section1_01 is separate;

    task body SUBSYSTEM01_TYPE is
    begin
       OUTER_LOOP:
       loop
          begin
          loop
             accept Wakeup;

             if EXIT_FLAG(SUBSYS_ID)  then
                exit OUTER_LOOP;
             end if;

             if BUS_OFFSET(SUBSYS_ID) = 0   then
                Section0_01;
             else
                Section1_01;
             end if;

             TASK_STATE(SUBSYS_ID) := IDLE;
          end loop;

          exception
             when SA_TYPES.EXIT_CONDITION   =>
                  ERROR_FLAG(SUBSYS_ID) := STOP_BLOCK;
             when NUMERIC_ERROR   =>
                  ERROR_FLAG(SUBSYS_ID) := MATH_ERROR;
             when OTHERS   =>
                  ERROR_FLAG(SUBSYS_ID) := UNKNOWN_ERROR;
          end;
       end loop OUTER_LOOP;
    end SUBSYSTEM01_TYPE;

end SUBSYSTEM01;


---------------
-- SCHEDULER --
---------------
with SA_TYPES;                     use SA_TYPES;
with SYSTEM_DATA;                  use SYSTEM_DATA;

package SCHEDULER_DATA is

    -- Scheduler Constants and Tables --

    type SUBSYSTEM_TYPE is (PERIODIC, ENABLED_PERIODIC, TRIGGERED_ANT,
                            TRIGGERED_ATR, TRIGGERED_SAF);

    type SUBSYSTEM_TYPE_ARRAY is array(1..NTASKS) of SUBSYSTEM_TYPE;
    type INITIAL_STATE_ARRAY  is array(1..NTASKS) of TASK_STATE_TYPE;

    SCHEDULER_ID            : constant INTEGER    := 0;
    OFFYE                   : constant RT_INTEGER := 28;
    NUMID                   : constant RT_INTEGER := 5;
    NUMENABLED              : constant RT_INTEGER := 0;
    NUMTRIGGERED            : constant RT_INTEGER := 0;
    NUMDSWRITERS            : constant RT_INTEGER := 0;
    DS_REGISTERS_EXT        : constant RT_INTEGER := 0;

    TASK_TYPE               : constant SUBSYSTEM_TYPE_ARRAY  :=
      (1..NTASKS => PERIODIC);
    INITIAL_TASK_STATE      : constant INITIAL_STATE_ARRAY   :=
      (1..NTASKS => IDLE);
    START_COUNT             : constant INTG_ARRAY(1..NTASKS) :=
      (1..NTASKS => 0);
    SCHEDULING_COUNT        : constant INTG_ARRAY(1..NTASKS) :=
      (1..NTASKS => 0);
    OUTPUT_COUNT            : constant INTG_ARRAY(1..NTASKS) :=
      (1..NTASKS => 0);
    SH_OFFSET               : constant INTG_ARRAY(1..NTASKS) :=
      (1..NTASKS => 33);
    SE_OFFSET               : constant INTG_ARRAY(1..NTASKS) :=
      (1..NTASKS => 5);
    NUMEXT                  : constant INTG_ARRAY(1..NTASKS) :=
      (1..NTASKS => 0);
    ID_EVENT                : constant INTG_ARRAY(1..NTASKS) :=
      (1..NTASKS => 0);
    LOC_EVENT               : constant INTG_ARRAY(1..NTASKS) :=
      (1..NTASKS => 0);
    DS_REGISTERS            : constant INTG_ARRAY(1..NTASKS) :=
      (1..NTASKS => 0);
    DS_OFFSET               : constant INTG_ARRAY(1..NTASKS) :=
      (1..NTASKS => 0);
    ID                      : constant INTG_ARRAY(1..NUMID)  :=
      (1, 1, 1, 1, 1);
    LOC                     : constant INTG_ARRAY(1..NUMID)  :=
      (10, 9, 7, 1, 11);
```

```
end SCHEDULER_DATA;


with SUBSYSTEM01;                  use SUBSYSTEM01;
with SA_UTILITIES;                 use SA_UTILITIES;
with SA_TYPES;                     use SA_TYPES;
with SCHEDULER_DATA;               use SCHEDULER_DATA;
with SYSTEM_DATA;                  use SYSTEM_DATA;
with SA_IO_DEFINITIONS;

package body SA_SCHEDULER is

    type TCB_TYPE is
       record
         TASK_TYPE           : SUBSYSTEM_TYPE;
         ENABLED             : BOOLEAN;
         START               : RT_INTEGER;
         START_COUNT         : RT_INTEGER;
         SCHEDULING_COUNT    : RT_INTEGER;
         ID_EVENT            : RT_INTEGER;
         LOC_EVENT           : RT_INTEGER;
         OUTPUT              : RT_INTEGER;
         OUTPUT_COUNT        : RT_INTEGER;
         SH_OFFSET           : RT_INTEGER;
         SE_OFFSET           : RT_INTEGER;
         NUMEXT              : RT_INTEGER;
         DS_OFFSET           : RT_INTEGER;
         DS_REGISTERS        : RT_INTEGER;
       end record;

    TSK                      : RT_INTEGER;
    OFFSE                    : RT_INTEGER;
    OFFSH                    : RT_INTEGER;
    OFFDS                    : RT_INTEGER;
    READY_COUNT              : RT_INTEGER;
    DISPATCH_COUNT           : RT_INTEGER;
    READY_QUEUE              : INTG_ARRAY(1..NTASKS);
    DISPATCH                 : BOOLEAN_ARRAY(1..NTASKS);
    TCB                      : array(1..NTASKS) of TCB_TYPE;


    procedure Queue_Task(NTSK : in RT_INTEGER) is
    begin
       READY_COUNT                  := READY_COUNT + 1;
       READY_QUEUE(READY_COUNT)     := NTSK;
       DISPATCH(NTSK)               := true;
       TASK_STATE(NTSK)             := RUNNING;
    end Queue_Task;
    pragma inline (Queue_Task);


    procedure Update_Outputs(NTSK : in RT_INTEGER) is
    begin
       BUS_OFFSET(NTSK)  := BUSIZE - BUS_OFFSET(NTSK);
    end Update_Outputs;
    pragma inline (Update_Outputs);


    procedure Signal_An_Error(NTSK : in RT_INTEGER) is
    begin
```

```
   if ERROR_FLAG(NTSK) = OK  then
      ERROR( NTSK, TIME_OVERFLOW );
   else
      ERROR( NTSK, ERROR_FLAG(NTSK) );
   end if;
end Signal_An_Error;


procedure Init_Scheduler is
begin
   for NTSK in 1..NTASKS  loop
      TCB(NTSK).TASK_TYPE        := TASK_TYPE(NTSK);
      TCB(NTSK).ENABLED          := false;
      TCB(NTSK).START            := START_COUNT(NTSK);
      TCB(NTSK).START_COUNT      := START_COUNT(NTSK);
      TCB(NTSK).SCHEDULING_COUNT := SCHEDULING_COUNT(NTSK);
      TCB(NTSK).ID_EVENT         := ID_EVENT(NTSK);
      TCB(NTSK).LOC_EVENT        := LOC_EVENT(NTSK);
      TCB(NTSK).OUTPUT           := OUTPUT_COUNT(NTSK);
      TCB(NTSK).OUTPUT_COUNT     := OUTPUT_COUNT(NTSK);
      TCB(NTSK).SH_OFFSET        := SH_OFFSET(NTSK);
      TCB(NTSK).SE_OFFSET        := SE_OFFSET(NTSK);
      TCB(NTSK).NUMEXT           := NUMEXT(NTSK);
      TCB(NTSK).DS_OFFSET        := DS_OFFSET(NTSK);
      TCB(NTSK).DS_REGISTERS     := DS_REGISTERS(NTSK);
      TASK_STATE(NTSK)           := INITIAL_TASK_STATE(NTSK);
      DISPATCH(NTSK)             := false;
      if TASK_TYPE(NTSK) = TRIGGERED_ATR or
         TASK_TYPE(NTSK) = TRIGGERED_SAF then
         BUS_OFFSET(NTSK) := 0;
      else
         BUS_OFFSET(NTSK) := BUSIZE;
      end if;
   end loop;
   DISPATCH_COUNT   := 0;
   READY_COUNT      := 0;
   READY_QUEUE(1)   := 0;
   SCHEDULER_STATUS := OK;
   INTERRUPT_COUNT  := 0;
end Init_Scheduler;


task body SCHEDULER_TYPE is
begin
  SCHEDULER_LOOP:
  loop
    INNER_LOOP:
    loop

      -- Scheduler Wakeup --

      accept Wakeup;

      INTERRUPT_COUNT := INTERRUPT_COUNT + 1;
      SA_IO_DEFINITIONS.STATUS_386.SCHEDULER_READY_INDEX := 0; -- False

      if SCHEDULER_STATUS /= OK  then
         ERROR(SCHEDULER_ID, SCHEDULER_STATUS);
         exit INNER_LOOP;
      end if;
```

```
-- System Input --

External_Input( BUS, UE_PTR, NUMIN, SCHEDULER_STATUS );

-- Task Scheduling --

for NTSK in reverse 1..NTASKS  loop

   case TASK_STATE(NTSK) is
      when IDLE =>

         case TCB(NTSK).TASK_TYPE is
            when PERIODIC =>
               if TCB(NTSK).START = 0 then
                  Queue_Task(NTSK);
                  Update_Outputs(NTSK);
                  TCB(NTSK).START  := TCB(NTSK).SCHEDULING_COUNT;
               else
                  TCB(NTSK).START  := TCB(NTSK).START - 1;
               end if;

            when ENABLED_PERIODIC =>
               if not TCB(NTSK).ENABLED then
                  TASK_STATE(NTSK) := BLOCKED;
               elsif TCB(NTSK).START = 0 then
                  Queue_Task(NTSK);
                  Update_Outputs(NTSK);
                  TCB(NTSK).START  := TCB(NTSK).SCHEDULING_COUNT;
               else
                  TCB(NTSK).START  := TCB(NTSK).START - 1;
               end if;

            when TRIGGERED_ANT =>
               if TCB(NTSK).START = 0 then
                  Queue_Task(NTSK);
                  Update_Outputs(NTSK);
                  TCB(NTSK).START  := 1;
               end if;

            when TRIGGERED_ATR =>
               if TCB(NTSK).OUTPUT = 0 then
                  Update_Outputs(NTSK);
                  TASK_STATE(NTSK) := BLOCKED;
               else
                  TCB(NTSK).OUTPUT := TCB(NTSK).OUTPUT - 1;
               end if;
               if TCB(NTSK).START = 0 then
                  Queue_Task(NTSK);
                  TCB(NTSK).OUTPUT := TCB(NTSK).OUTPUT_COUNT;
                  TCB(NTSK).START  := 1;
               end if;

            when TRIGGERED_SAF =>
               if TCB(NTSK).OUTPUT = 0 then
                  Update_Outputs(NTSK);
                  TASK_STATE(NTSK) := BLOCKED;
               end if;
               if TCB(NTSK).START = 0 then
                  Queue_Task(NTSK);
            TCB(NTSK).OUTPUT := 0;
            TCB(NTSK).START  := 1;
         end if;

      when OTHERS =>
         null;
   end case;

when RUNNING =>

   case TCB(NTSK).TASK_TYPE is
      when PERIODIC =>
         if TCB(NTSK).START > 0 then
            TCB(NTSK).START  := TCB(NTSK).START - 1;
         else
            Signal_An_Error(NTSK);
            exit INNER_LOOP;
         end if;

      when ENABLED_PERIODIC =>
         if TCB(NTSK).START > 0 then
            TCB(NTSK).START  := TCB(NTSK).START - 1;
         else
            Signal_An_Error(NTSK);
            exit INNER_LOOP;
         end if;

      when TRIGGERED_ANT =>
         if TCB(NTSK).START = 0 then
            Signal_An_Error(NTSK);
            exit INNER_LOOP;
         end if;

      when TRIGGERED_ATR =>
         if TCB(NTSK).OUTPUT > 0 then
            TCB(NTSK).OUTPUT := TCB(NTSK).OUTPUT - 1;
         else
            Signal_An_Error(NTSK);
            exit INNER_LOOP;
         end if;

      when TRIGGERED_SAF =>
         if ERROR_FLAG(NTSK) /= 0 then
            Signal_An_Error(NTSK);
            exit INNER_LOOP;
         end if;

      when OTHERS =>
         null;
   end case;

when BLOCKED =>

   case TCB(NTSK).TASK_TYPE is
      when ENABLED_PERIODIC =>
         if TCB(NTSK).ENABLED then
            Queue_Task(NTSK);
            Update_Outputs(NTSK);
            TCB(NTSK).START  := TCB(NTSK).SCHEDULING_COUNT;
         end if;
```

```
            when TRIGGERED_ATR =>
               if TCB(NTSK).START = 0 then
                  Queue_Task(NTSK);
                  TCB(NTSK).OUTPUT := TCB(NTSK).OUTPUT_COUNT;
                  TCB(NTSK).START  := 1;
               end if;

            when TRIGGERED_SAF =>
               if TCB(NTSK).START = 0 then
                  Queue_Task(NTSK);
                  TCB(NTSK).OUTPUT := 0;
                  TCB(NTSK).START  := 1;
               end if;

            when OTHERS =>
               null;
         end case;

   end case;
end loop;

-- System Output --

for J in 1..NUMOUT    loop
   BUS(J+OFFYE) := BUS( LOC(J)+BUS_OFFSET(ID(J)) );
end loop;

External_Output ( BUS, YE_PTR, NUMOUT, SCHEDULER_STATUS );

-- Task Input Sample and Hold --

for I in reverse 1..READY_COUNT  loop
   TSK   := READY_QUEUE(I);
   OFFSE := TCB(TSK).SE_OFFSET;
   OFFSH := TCB(TSK).SH_OFFSET - OFFSE;
   for J in OFFSE+1..OFFSE+TCB(TSK).NUMEXT   loop
      BUS(J+OFFSH) := BUS( LOC(J)+BUS_OFFSET(ID(J)) );
   end loop;
end loop;

-- Clear Ready Queue --

if READY_QUEUE(1) > DISPATCH_COUNT then
   DISPATCH_COUNT := READY_QUEUE(1);
end if;
READY_COUNT    := 0;
READY_QUEUE(1) := 0;

-- Task Dispatching --

for NTSK in 1..DISPATCH_COUNT  loop
   if DISPATCH(NTSK) then
      DISPATCH(NTSK) := false;
      case NTSK is
         when 1       =>  SUBSYSTEM_1.Wakeup;
         when OTHERS  =>  null;
      end case;
   end if;
end loop;
```

```
          DISPATCH_COUNT := 0;

          INTERRUPT_COUNT := INTERRUPT_COUNT - 1;
          SA_IO_DEFINITIONS.STATUS_386.SCHEDULER_READY_INDEX := 1;   -- True

        end loop INNER_LOOP;

        exit;

      end loop SCHEDULER_LOOP;

      -- Shutdown --

      accept Quit;

    end SCHEDULER_TYPE;
end SA_SCHEDULER;


----------
-- MAIN --
----------
with SUBSYSTEM01;                  use SUBSYSTEM01;
with SA_MPC_INTERRUPT;             use SA_MPC_INTERRUPT;
with SA_IO_DEFINITIONS;            use SA_IO_DEFINITIONS;
with SA_SCHEDULER;                 use SA_SCHEDULER;
with SA_UTILITIES;                 use SA_UTILITIES;
with SYSTEM_DATA;                  use SYSTEM_DATA;

procedure AQUA_SYS is

    ------------------
    -- COMMAND_TASK --
    ------------------
    task type COMMAND_TASK_TYPE is
        pragma PRIORITY (LOWEST_PRIORITY);
    end COMMAND_TASK_TYPE;

    type COMMAND_TASK_POINTER is access COMMAND_TASK_TYPE;
    COMMAND_TASK  : COMMAND_TASK_POINTER;

    task body COMMAND_TASK_TYPE is
    begin
        while STATUS_386.TASK_COMMAND /= CONVERT_INPUT_DATA    loop
          for WAIT_CNT in 1..10    loop
             null;
          end loop;
        end loop;
        SCHEDULER_STATUS := STOPPED;
        SCHEDULER.Wakeup;
    end COMMAND_TASK_TYPE;

    procedure Shutdown_The_Tasks is
    begin
        for NTSK in 1..NTASKS  loop
          EXIT_FLAG(NTSK) := true;
        end loop;
        SUBSYSTEM_1.Wakeup;
    end Shutdown_The_Tasks;
```

```
begin

    -- Create Subsystem Tasks --

    SUBSYSTEM_1  := new SUBSYSTEM01_TYPE;

    -- Start Scheduler --

    Init_Scheduler;
    SCHEDULER := new SCHEDULER_TYPE;

    -- User Initialization --

    Implementation_Specific_Initialize (BUS, UE_PTR, NUMIN, YE_PTR, NUMOUT,
                                        SCHEDULER_STATUS, SCHEDULER_FREQ);

    if  SCHEDULER_STATUS = OK  then
        COMMAND_TASK := new COMMAND_TASK_TYPE;
    else
        SCHEDULER.Wakeup;
    end if;

    -- User Termination --

    SCHEDULER.Quit;

    if  STATUS_386.TASK_COMMAND = CONVERT_INPUT_DATA   then
        Implementation_Specific_Terminate;
    else
        STATUS_386.ERRORTYPE := 1; -- Reboot controller
    end if;

    Shutdown_The_Tasks;

end AQUA_SYS;



separate (SUBSYSTEM01)
    procedure Section0_01 is
    begin

    ------------------------------------  General Nested Expression
    -- {CONTROLLER.VOLT TO TEMP.6}
        BUS(12) := 60.0 + 8.0*BUS(24);
    ------------------------------------  Summing Junction
    -- {CONTROLLER..7}
        BUS(13) := -BUS(12) + BUS(23);
    ------------------------------------  Dead Band
    -- {CONTROLLER.TEMP ERROR DEADBAND.16}
        BUS(14) := BUS(13) - 3.0;
        if BUS(14) < 0.0  then
           BUS(14) := BUS(13) + 3.0;
           if BUS(14) > 0.0  then
              BUS(14) := 0.0;
           end if;
        end if;
    ------------------------------------  General Logical Expression
    -- {CONTROLLER.RAISE TEMPERATURE.5}
        if BUS(14)>0.0 then
```

```
            BUS(15)  := 1.0;
        else
            BUS(15)  := 0.0;
        end if;
    ------------------------------------  OR   Logical Block
    -- {CONTROLLER..95}
        if  BUS(25) > 0.0 or  BUS(26) > 0.0  then
            BUS(16)  := 1.0;
        else
            BUS(16)  := 0.0;
        end if;
    ------------------------------------  NOT  Logical Block
    -- {CONTROLLER..96}
        if  BUS(16) > 0.0  then
            BUS(17)  := 0.0;
        else
            BUS(17)  := 1.0;
        end if;
    ------------------------------------  AND  Logical Block
    -- {CONTROLLER..97}
        if  BUS(15) > 0.0 and BUS(17) > 0.0  then
            BUS(18)  := 1.0;
        else
            BUS(18)  := 0.0;
        end if;
    ------------------------------------  Data Conversion
    -- {CONTROLLER.LOG_INT CONVERSION.26}
        BUS(19)  := RT_FLOAT(RT_INTEGER(BUS(18)));
    ------------------------------------  Gain Block
    -- {CONTROLLER..92}
        BUS(20)  := 10.0 * BUS(19);
    ------------------------------------  Gain Block
    -- {CONTROLLER.LIGHT.99}
        BUS(21)  := BUS(27);
    ------------------------------------  General Nested Expression
    -- {CONTROLLER.WATER LEVEL CONSTANT.98}
        BUS(22)  := 3.0;

    end Section0_01;



separate (SUBSYSTEM01)
    procedure Section1_01 is
    begin

    ------------------------------------  General Nested Expression
    -- {CONTROLLER.VOLT TO TEMP.6}
        BUS(1)  := 60.0 + 8.0*BUS(24);
    ------------------------------------  Summing Junction
    -- {CONTROLLER..7}
        BUS(2)  := -BUS(1) + BUS(23);
    ------------------------------------  Dead Band
    -- {CONTROLLER.TEMP ERROR DEADBAND.16}
        BUS(3)  := BUS(2) - 3.0;
        if BUS(3) < 0.0  then
           BUS(3)  := BUS(2) + 3.0;
           if BUS(3) > 0.0  then
              BUS(3)  := 0.0;
           end if;
    end if;
------------------------------------  General Logical Expression
-- {CONTROLLER.RAISE TEMPERATURE.5}
    if BUS(3)>0.0 then
        BUS(4) := 1.0;
    else
        BUS(4) := 0.0;
    end if;
------------------------------------  OR   Logical Block
-- {CONTROLLER..95}
    if  BUS(25) > 0.0 or  BUS(26) > 0.0  then
        BUS(5) := 1.0;
    else
        BUS(5) := 0.0;
    end if;
------------------------------------  NOT  Logical Block
-- {CONTROLLER..96}
    if  BUS(5) > 0.0  then
        BUS(6) := 0.0;
    else
        BUS(6) := 1.0;
    end if;
------------------------------------  AND  Logical Block
-- {CONTROLLER..97}
    if  BUS(4) > 0.0 and BUS(6) > 0.0  then
        BUS(7) := 1.0;
    else
        BUS(7) := 0.0;
    end if;
------------------------------------  Data Conversion
-- {CONTROLLER.LOG_INT CONVERSION.26}
    BUS(8) := RT_FLOAT(RT_INTEGER(BUS(7)));
------------------------------------  Gain Block
-- {CONTROLLER..92}
    BUS(9) := 10.0 * BUS(8);
------------------------------------  Gain Block
-- {CONTROLLER.LIGHT.99}
    BUS(10) := BUS(27);
------------------------------------  General Nested Expression
-- {CONTROLLER.WATER LEVEL CONSTANT.98}
    BUS(11) := 3.0;

end Section1_01;
```

Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.

4. Title and Subtitle

Software Technology Testbed "SoftPanel" Prototype Final Report

5. Report Date

February 1991

6. Performing Organization Code

7. Author(s)

GDI

8. Performing Organization Report No.

10. Work Unit No.

9. Performing Organization Name and Address

General Digital Industries, Inc.
6705 Odyssey Drive
Huntsville, AL 35806

11. Contract or Grant No.

NAS8-37680

12. Sponsoring Agency Name and Address

13. Type of Report and Period Covered

14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

Provides final report of activities performed under this contract including work on Touch Panel Testbed technology, software standards, and Independent Software Analysis of Space Station Freedom Software.

17. Key Words (Suggested by Author(s))
Soft Panel
SMADR
MASM
Aquarium System
AG

18. Distribution Statement

Unclassified - Unlimited

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of pages	22. Price
Unclassified	---	68	